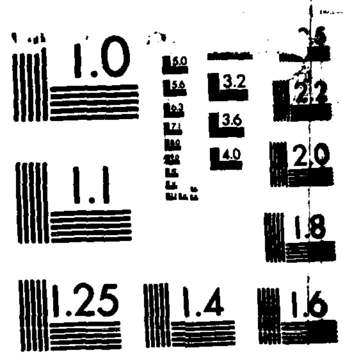
AD-A164 356 REDUCTION IN BANDWIDTH BY USING VARIABLE LENGTH CODES (U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA S AKINSEL DEC 85 1/2

UNCLASSIFIED F/G 9/4 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(2)

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A164 356

DTIC ELECTE FEB 2 0 1986 S D D

THESIS

REDUCTION IN BANDWIDTH
BY USING VARIABLE LENGTH CODES

by

Serdar Akinsel

December 1985

Thesis Advisor: R. W. Hamming

Approved for public release; distribution is unlimited.

AD-A164356

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release; distribution is unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
Naval Postgraduate School

6b. OFFICE SYMBOL *(If applicable)*
Code 62

7a. NAME OF MONITORING ORGANIZATION
Naval Postgraduate School

6c. ADDRESS *(City, State, and ZIP Code)*
Monterey, California 93943-5100

7b. ADDRESS *(City, State, and ZIP Code)*
Monterey, California 93943-5100

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State, and ZIP Code)*

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
REDUCTION IN BANDWIDTH BY USING VARIABLE LENGTH CODES

12. PERSONAL AUTHOR(S)
Akinsel, Serdar

13a. TYPE OF REPORT
Master's Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Year, Month, Day)*
1985 December

15. PAGE COUNT
113

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Huffman Codes, Reduction in Variance, Increase in Mean Time, Reduction in Bandwidth, Decoding of Variable Length Codes, Time Delay.

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A method of coding an ensemble of messages of a finite number of symbols is developed. Minimizing the average number of coding digits per message by using Huffman coding can result in a large variance. This is a problem because a large variance requires a large buffer and also creates more time delay during transmission and decoding respectively for on-line communication.

This research examines modified Huffman codes for the purpose of finding a way to reduce the variance. The effective parameters which give the lower variance modified Huffman codes are obtained. The buffer requirements and the reduction of the bandwidth to forward messages in an on-line communication is investigated. A possible design for a practical system is presented for using the modified Huffman codes.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
R. W. Hamming

22b. TELEPHONE *(Include Area Code)*
(408) 646-2655

22c. OFFICE SYMBOL
52Hg

Approved for public release; distribution is unlimited.

Reduction in Bandwidth by Using Variable Length Codes

by

Serdar Akinsel
Lt.Jg., Turkish Navy
B.S., Turkish Naval Academy, 1979

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1985

Author: S. Akinsel
Serdar Akinsel

Approved by: R.W. Hamming
R.W. Hamming, Thesis Advisor

H. Fredricksen
H. Fredricksen, Second Reader

Harriett Rigas
Harriett B. Rigas, Chairman,
Department of Electrical and Computer Engineering

John N. Dyer
John N. Dyer,
Dean of Science and Engineering

ABSTRACT

A method of coding an ensemble of messages of a finite number of symbols is developed. Minimizing the average number of coding digits per message by using Huffman coding can result in a large variance. This is a problem because a large variance requires a large buffer and also creates more time delay during transmission and decoding respectively for on-line communication.

This research examines modified Huffman codes for the purpose of finding a way to reduce the variance. The effective parameters which give the lower variance modified Huffman codes are obtained. The buffer requirements and the reduction of the bandwidth to forward messages in an on-line communication is investigated. A possible design for a practical system is presented for using the modified Huffman codes.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The author wishes to gratefully acknowledge his thesis advisor, Prof. R. W. Hamming, for suggesting the basis of this thesis, and for his invaluable advice and guidance during the course of this work.

The author would also like to express his appreciation to Prof. H. Fredricksen for his constructive critism as a second reader, and to Prof. Bruce Mclennan for the use of his program for Huffman coding (and his modification of it) to obtain different variable length codes.

A special thanks is due to the author's wife Mrs. Seher Akinsel, for helping me type this thesis and especially for supporting the author's study at the Naval Postgraduate School.

I. THE INTRODUCTION

The two main problems of representing the source alphabet symbols in terms of another system of symbols (the encoding process) are the following:

1. The altered symbols could be decoded incorrectly.
2. For the sake of efficiency the source symbols should be represented in a minimal form.

One coding process is to encode the source data into the binary digits (bits) which consists of 0's and 1's. Modern military communication systems are increasingly adopting the digital method of transmitting data. In addition to its simplicity part of the reason for the use of digital transmission is that it is more reliable than is analog transmission. Another reason that modern systems use digital methods is that integrated circuits are now very cheap and provide a powerful method for flexibly and reliably processing and transforming digital signals.

Dealing with digital transmission, shorter messages maximize the data transfer rate but also cause minimization of the redundancy of the source and hence vulnerability to errors. Variable length codes used to encode the source data drives the reduction in the source redundancy.

The Huffman code is clearly a variable length code. It takes advantage of the high frequency occurence of some letters in the source alphabet by assigning them short bit sequences. On the other hand the low frequency occurence of source symbols are assigned long bit sequences. [Ref. 1]

A. HUFFMAN CODING

Huffman encoding, devised by David A. Huffman, has the property of being a minimum redundancy encoding; that is, among all variable length binary encodings having the prefix property, that no complete symbol is the prefix of some

other symbol, this encoding has the lowest average number of binary digits used per letter of the original message, assuming that the message is made up of letters independently chosen, each with its probability given. [Ref. 2]

The underlying idea of the Huffman coding procedure is to repeatedly reduce a code to an equivalent problem with one less code symbol. In more detail, the two least probable symbols are merged into a single symbol whose probability is the sum of the two original probabilities and then this new symbol is inserted into its proper (ordered) position. As an example of Huffman encoding suppose we have a source alphabet of six symbols, with the given probabilities of occurence. See (Table 1).

TABLE 1
SOURCE ALPHABET AND ITS PROBABILITIES

SYMBOL	PROBABILITIES
S1	0.4
S2	0.2
S3	0.2
S4	0.1
S5	0.05
S6	0.05

From Table 1 it can be observed that the sum of the probabilities is equal to one. If the symbols don't have the probabilities in decreasing order, they should be arranged in this way. The coding process can be done according to the following procedure.

To obtain the first reduction from the original n symbols to n-1 symbols combine the two least probable symbols of the source alphabet into a single symbol, whose probability is equal to the sum of the two corresponding probabilities. See (Figure 1.1).

Symbol	Prob.	Prob.
S1	0.4	0.4
S2	0.2	0.2
S3	0.2	0.2
S4	0.1	0.1
S5	0.05	0.1
S6	0.05	
	Original	First Reduction

Figure 1.1 The First Reduction.

The repetition of this reduction is to be continued until only two symbols remain. See (Figure 1.2). As in the original, in each reduction the probability summation equal to one is kept.

By giving the two symbols in the fourth reduction the values 0 and 1, and proceeding backwards to the left, the assignments for the original code words can be accomplished. Going backwards, one of these symbols has to be expanded into two symbols. By assigning a second digit 0 for one of them and 1 for the the other, this splitting process is continued until one comes back to the original symbols. Figure 1.3 shows the first three splitting processes and their respective assigned code words. The 0 and 1's in the parentheses are the assigned code words. The final code words for this example are given in Table 2.

Symbol	Prob.	Prob.	Prob.	Prob.	Prob.
S1	0.4	0.4	0.4	0.4	0.6
S2	0.2	0.2	0.2	0.4	0.4
S3	0.2	0.2	0.2	0.2	
S4	0.1	0.1	0.2		
S5	0.05	0.1			
S6	0.05				
	Original	First Reduc.	Second Reduc.	Third Reduc.	Fourth Reduc.

Figure 1.2 The Reduction Process.

Symbol	Prob.	Prob.	Prob.
S1	0.4(1)	0.4(1)	0.6(0)
S2	0.2(01)	0.4(00)	0.4(1)
S3	0.2(000)	0.2(01)	
S4	0.2(001)		
S5			
S6			
	Third Splitting	Second Splitting	First Splitting

Figure 1.3 The First Three Splitting Processes.

B. MODIFICATION OF HUFFMAN CODING

The procedure given in section A was accomplished by merging states at the bottom of the list of ordered probabilities. The code word lengths which were assigned to the symbols of the above example were (1,2,3,4,5,5) as shown in Table 2. The average code length is given by

$$L = 0.4(1) + 0.2(2) + 0.2(3) + 0.1(4) + 0.05(5) + 0.05(5)$$

$$L = 2.3$$

and the variance is given by

$$V = 0.4(1-2.3)^2 + 0.2(2-2.3)^2 + 0.2(3-2.3)^2 + 0.1(4-2.3)^2 + 0.05(5-2.3)^2 + 0.05(5-2.3)^2 = 1.81$$

On the other hand, if the combined symbols are placed as high as possible in the list of ordered probabilities the code lengths obtained will be (2,2,2,3,4,4). For this second encoding the reductions are given in Figure 1.4. The first three splitting processes and final assignments of the source symbols are shown in Figure 1.5 and in Table 3 respectively.

The average code length is now given by

$$L = 0.4(2) + 0.2(2) + 0.2(2) + 0.1(3) + 0.05(4) + 0.05(4)$$

$$L = 2.3$$

and the variance is given by

$$V = 0.4(2-2.3)^2 + 0.2(2-2.3)^2 + 0.2(2-2.3)^2 + 0.1(3-2.3)^2 + 0.05(4-2.3)^2 + 0.05(4-2.3)^2 = 0.41$$

Obviously the variability of the second assignment is lower than that of the first code. The result of moving merged symbols to high positions will result in the production of codes of lower variance. [Ref. 1: page 68]

To obtain codes of low variance as a modification of Huffman coding, three different parameters (N,K,E) are defined to describe the position where the combined symbol is to be placed. These three parameters, two of which were proposed in [Ref. 3], make use of what appears to be an optimal (in the sense minimizing variance) procedure of shifting the combined symbols higher than where they belong in the ordered probability listing. The definitions and examples of these parameters are given below.

TABLE 2

FINAL ASSIGNMENTS OF THE CODE WORDS

SYMBOL	CODE WORDS
S1	1
S2	01
S3	000
S4	0010
S5	00110
S6	00111

Symbol	Prob.	Prob.	Prob.	Prob.	Prob.
S1	0.4	0.4	0.4	0.4	0.6
S2	0.2	0.2	0.2	0.4	0.4
S3	0.2	0.2	0.2	0.2	
S4	0.1	0.1	0.2		
S5	0.05	0.1			
S6	0.05				
	Original	First Reduc.	Second Reduc.	Third Reduc.	Fourth Reduc.

Figure 1.4 The Reduction Process for the Second Encoding.

1. <u>The Parameter N</u>

N is defined as an integer which is used to move the merged symbols to relatively higher positions than would be normally done. If N is set to 2, combined symbols are moved two positions higher than they would normally appear in the list of probabilities. Setting N equal to 0, the original

Symbol	Prob.	Prob.	Prob.
S1	0.4(00)	0.4(1)	0.6(0)
S2	0.2(01)	0.4(00)	0.4(1)
S3	0.2(10)	0.2(01)	
S4	0.2(11)		
S5			
S6			
	Third Splitting	Second Splitting	First Splitting

Figure 1.5 Splitting Processes for the Second Encoding.

TABLE 3
THE FINAL CODE WORDS FOR THE SECOND ENCODING

SYMBOL	CODE WORDS
S1	00
S2	10
S3	11
S4	011
S5	0100
S6	0101

Huffman encoding given in Table 1 can be obtained. Figure 1.6 demonstrates the first reduction of the modified Huffman coding for the example given in the previous section when N is set to 2. When the second reduction is performed the last symbols in the list are combined. In the example, the last two symbols of the first reduction (0.2 and 0.1) are

merged and the probability assigned to the combined symbol is 0.3. This merged new symbol is then placed at the top of the list.

Symbol	Prob.	Prob.
S1	0.4	0.4
S2	0.2	0.2
S3	0.2	0.1
S4	0.1	0.2
S5	0.05	0.1
S6	0.05	
	Original	First Reduction

Figure 1.6 Modified Huffman Coding for N = 2.

2. <u>The Parameter K</u>

The second parameter, K, is a number used to multiply the probability sum of each merged entry. This parameter generally causes the merged entry to appear in higher position than it would appear normally in the original Huffman coding. The original Huffman code is obtained by setting K to 1. Setting K equal to 3, for example, multiplies the probability of the combined entry by 3 and then puts it where it would normally appear. Of course now the probabilities no longer add to 1. The first reduction of the Huffman coding given in the previous section can be modified as shown in Figure 1.7.

3. <u>The Parameter E</u>

The third parameter, E, is a real number added to the sum of the probabilities of the merged entries. As far as the relative positions in the list of symbols are concerned, E has the same effect as K and N. The merged symbol is placed in the location where it would normally

Symbol	Prob.	Prob.
S1	0.4	0.4
S2	0.2	0.3
S3	0.2	0.2
S4	0.1	0.2
S5	0.05	0.1
S6	0.05	
	Original	First Reduction

Figure 1.7 Modified Huffman Coding for K = 3.

appear as if the result was the correct probability. Of course, again the probabilities do not sum to 1. The original Huffman coding is produced when E is set to 0. Figure 1.8 shows the first reduction of the modified Huffman coding for the example given in the previous section when E is set to 0.15.

Symbol	Prob.	Prob.
S1	0.4	0.4
S2	0.2	0.25
S3	0.2	0.2
S4	0.1	0.2
S5	0.05	0.1
S6	0.05	
	Original	First Reduction

Figure 1.8 Modified Huffman Coding for E = 0.15.

C. THE NORMALIZATION PROCESS

During the modification of the Huffman coding process, the requirement for the summation of the probabilities to be equal to one was not considered except for the parameter N. For the other two parameters (E,K) a probability sum equal to one can be retained by normalizing the list of ordered probabilities at each reduction stage during the modification process. To observe the effect of this normalization we first continue the reduction process and find the code words produced when the parameter E is set to 0.15 without normalizing. See (Figure 1.9). The splitting process and the final code words are given in Figure 1.10 and in Table 4, respectively.

On the other hand, if the normalization is applied at each reduction stage, the resulting reduction and splitting processes are as given in Figure 1.11 and Figure 1.12. Finally the code words after normalization are as shown in Table 5. The normalized probabilities at each reduction stage were obtained by dividing each probability by 1.15 (normalization parameter) for this particular example.

Table 4 and Table 5 emphasize that the same code words would be obtained either with or without normalization. The effect of normalization is only to decrease the probabilities at each reduction stage to a smaller number. But if the normalization parameter is a large number then the order of probabilities at reduction processes could be slightly different, resulting in slightly different code words. Clearly since the same code words are obtained there is no need to perform the work required for the normalization.

Symbol	Prob.	Prob.	Prob.	Prob.	Prob.
S1	0.4	0.4	0.45	0.6	1.0
S2	0.2	0.25	0.4	0.45	0.6
S3	0.2	0.2	0.25	0.4	
S4	0.1	0.2	0.2		
S5	0.05	0.1			
S6	0.05				
	Original	First Reduc.	Second Reduc.	Third Reduc.	Fourth Reduc.

Figure 1.9   The Reduction Process when E = 0.15.

Symbol	Prob.	Prob.	Prob.	Prob.	Prob.
S1	0.4(01)	0.4(01)	0.45(00)	0.6(1)	1.0(0)
S2	0.2(11)	0.25(10)	0.4(01)	0.45(00)	0.6(1)
S3	0.2(000)	0.2(11)	0.25(10)	0.4(01)	
S4	0.1(001)	0.2(000)	0.2(11)		
S5	0.05(100)	0.1(001)			
S6	0.05(101)				
	Final Split.	Fourth Split.	Third Split.	Second Split.	First Split.

Figure 1.10   The Splitting Process when E = 0.15.

TABLE 4

THE FINAL CODE WORDS WHEN E = 0.15

SYMBOL	CODE WORDS
S1	01
S2	11
S3	000
S4	001
S5	100
S6	101

Symbol	Prob.	Prob.	Norm. Prob.	Prob.	Norm. Prob.	Prob.	Norm. Prob.	Prob.	Norm. Prob.
S1	0.4	0.4	0.348	0.410	0.357	0.49	0.426	0.724	0.63
S2	0.2	0.25	0.217	0.348	0.303	0.357	0.310	0.426	0.37
S3	0.2	0.2	0.174	0.217	0.189	0.303	0.264		
S4	0.1	0.2	0.174	0.174	0.151				
S5	0.05	0.1	0.087						
S6	0.05								
	Orig.	First Norm. Reduction		Second Norm. Reduction		Third Norm. Reduction		Fourth Norm. Reduction	

Figure 1.11 The Normalized Reduction Process when E = 0.15.

Sym.	Prob.	Prob.	Prob.	Prob.	Prob.
S1	0.4(01)	0.348(01)	0.357(00)	0.426(1)	0.630(0)
S2	0.2(11)	0.217(10)	0.303(01)	0.310(00)	0.370(1)
S3	0.2(000)	0.174(11)	0.189(10)	0.264(01)	
S4	0.1(001)	0.174(000)	0.151(11)		
S5	0.05(100)	0.087(001)			
S6	0.05(101)				
	Final Split	Fourth Split.	Third Split.	Second Split.	First Split.

Figure 1.12 The Normalized Splitting Process (E = 0.15).

TABLE 5

THE NORMALIZED FINAL CODE WORDS (E = 0.15)

SYMBOL	CODE WORDS
S1	01
S2	11
S3	000
S4	001
S5	100
S6	101

II. MODIFICATION OF HUFFMAN CODING FOR A PARTICULAR ALPHABET

A. A PARTICULAR ALPHABET

Huffman coding produces the code with the minimum average code length. Here we propose to find a practical modified variable length code for the Turkish alphabet to minimize the average code length and also minimize the variance using techniques involving the parameters introduced in the previous chapter.

For the following two reasons the use of the Turkish alphabet was not possible:

1. The exact probabilities of the Turkish alphabet are not known.
2. Some of the letters in Turkish alphabet are not available on the keyboard.

Therefore, it was determined to use the same alphabet, given in [Ref. 3]. This alphabet consists of 47 characters with common usage letters, numbers (0-9) and special symbols for the use of the on-line communication.

Two Turkish magazine articles [Refs. 4,5] were used to obtain the approximate frequencies of occurences of symbols of the Turkish alphabet. A Fortran language program and Statistical Analysis System (SAS) package program [Ref. 6], was executed to determine the probabilities as was done in [Ref. 3]. These magazine articles, the Fortran language program and SAS program appear in Appendix A.

Table 6 contains the data taken from the output of these programs. The characters with their probabilities in descending order are given in Table 7.

TABLE 6

SYMBOL CHARACTERISTICS OF THE PARTICULAR ALPHABET

SYMBOL	FREQUENCY	CUM FREQ	PERCENT	CUM PERCENT
.	182	182	1.017	1.017
(	12	194	0.067	1.084
)	15	209	0.084	1.168
'	11	220	0.061	1.229
-	3	223	0.017	1.246
space	2387	2610	13.339	14.585
,	219	2829	1.224	15.809
?	1	2830	0.006	15.814
:	6	2836	0.034	15.848
;	29	2865	0.162	16.010
"	20	2885	0.112	16.122
A	1687	4572	9.427	25.549
B	337	4909	1.883	27.432
C	293	5202	1.637	29.070
D	628	5830	3.509	32.579
E	1423	7253	7.952	40.531
F	64	7317	0.358	40.889
G	391	7708	2.185	43.073
H	104	7812	0.581	43.655
I	1884	9696	10.528	54.183
J	8	9704	0.045	54.227
K	691	10395	3.861	58.089
L	918	11313	5.130	63.219
M	527	11840	2.945	66.164
N	1183	13023	6.611	72.775
O	476	13499	2.660	75.434
P	123	13622	0.687	76.122
R	1089	14711	6.085	82.207
S	713	15424	3.984	86.192
T	575	15999	3.213	89.405
U	924	16923	5.163	94.568
V	156	17079	0.872	95.440
W	7	17086	0.039	95.479
X	1	17087	0.006	95.485
Y	480	17567	2.682	98.167
Z	177	17744	0.989	99.156
0	35	17779	0.196	99.352
1	24	17803	0.134	99.486
2	16	17819	0.089	99.575
3	13	17832	0.073	99.648
4	12	17844	0.067	99.715
5	15	17859	0.084	99.799
6	8	17867	0.045	99.844
7	5	17872	0.028	99.871
8	13	17885	0.073	99.944
9	10	17895	0.056	100.000

TABLE 7

SYMBOL PROBABILITIES

SYMBOL	PROBABILITY	SYMBOL	PROBABILITY
space	0.13339	F	0.00358
I	0.10528	0	0.00196
A	0.09427	'	0.00162
E	0.07952	1	0.00134
N	0.06611	"	0.00112
R	0.06085	2	0.00089
U	0.05163	)	0.00084
L	0.05130	5	0.00084
S	0.03984	3	0.00073
K	0.03861	8	0.00073
D	0.03509	(	0.00067
T	0.03213	4	0.00067
M	0.02945	;	0.00061
Y	0.02682	9	0.00056
O	0.02660	J	0.00045
G	0.02185	6	0.00045
B	0.01883	W	0.00039
C	0.01637	:	0.00034
,	0.01224	7	0.00028
.	0.01017	-	0.00017
Z	0.00989	?	0.00006
V	0.00872	X	0.00006
P	0.00687	Q	0.00000
H	0.00581		

B. EXPERIMENTAL PARAMETER

The three different parameters N, K, E introduced in the first chapter were investigated with this particular alphabet to obtain lower variance codes than the original Huffman code. Because of the size of alphabet the modification process was not performed manually. A program written in a List programming language (LISP) was used, to produce the encoding. The output of the program gives the code words with their average lengths and the corresponding variances. This program is given in Appendix B [Refs. 7,8]. This program was run employing these three parameters N, K and E. The original Huffman coding can be obtained for this particular alphabet as before by setting parameters N and E to 0 and K to 1. According to the discussion in the previous chapter the normalization process was not considered necessary. These parameters were tested separately in order to find which parameter gives the best codes when each is used independently. The following three basic steps were performed.

<u>Step1</u>. For parameter N, the program was executed 31 times. N was selected as each integer value from 0 to 30. While doing this the other two parameters, E and K, were fixed at 0 and at 1 respectively in order not to affect N.

<u>Step2</u>. For parameter K the program was run 2072 times with K set equal to a sequence of rational numbers from 1. to 275. The average lengths and corresponding variances of the codes for each K were obtained. In this step parameters N and E were each set to zero for testing only the parameter K.

<u>Step3</u>. For the parameter E the program was executed 2273 times with the values ranging from 0.0 to 0.3. The various values used yielded different codes with their mean times and variances. In order not to affect E, the other two parameters, N and K was set to 0 and to 1, respectively, while running the LISP program.

During the modification process applied to this particular alphabet, average code lengths and variances of the encoding for different values of the parameters were obtained. As far as unique mean times and variances are concerned, 24, 62, 251 unlike codes were obtained by the

choices for the parameters N, K and E respectively. For some values of the parameters, the resulting mean times and variances of the encoding are the same. Since the combined symbols were positioned as high as they could go in the reduction processes, the same mean times and variances were obtained after some certain values of the parameters. For example, for this particular alphabet for $N \geq 29$, $K \geq 272.84$ and $E \geq 0.264$, the same average code lengths and variances were obtained equal to 5.08843 and 0.08061 in each of the three different steps.

The preceeding three steps also emphasize the fact that among all choices of the parameters, more codes are generated by using the third parameter E. Since this parameter E can be any real number, it can be adjusted so that the merged symbols do not move to higher positions in some of the reduction processes for some values of E but the merged symbols do move for some other values of E. On the other hand, by using some large values for N and K the merged symbols typically are brought to higher positions in the beginning of the reduction process. As more flexibility in the reduction processes is possible by the choice of values for E, more codes can be obtained using the parameter E, since the merged symbols do not always move to higher locations.

The different average lengths and variances, of the modified Huffman codes obtained with different parameter values of N, K, and E values are given in Tables 8, 9 and 10 respectively. The average length and variance obtained by setting N and E equal to 0, and K equal to 1, corresponds to the original Huffman code for this particular alphabet. The different values of the parameters given in these tables represent the minimum values for the given parameter which result in a given mean time and variance. For instance, all the codes using E = 0.00011 up to E = 0.00033 have the same average length and variance thus E = 0.00011 appears in

Table 10. To be able to determine a good experimental parameter, graphs which contain mean times on the horizontal axis and variances on the vertical axis were plotted for each parameter, separately. These graphs are shown in Figure 2.1, 2.2 and 2.3. The second and the third columns of Tables 8, 9 and 10 were used for the data in these figures. When these three graphs are compared with each other, the minimum variances for the corresponding average code lengths (dashed lines) were found for the codes due to the parameters N, K and E.

C. ASSIGNMENT OF THE CODES FOR THE PARAMETER E

According to the discussion in the previous section, E was chosen as a more robust parameter than the parameters N and K for modifying Huffman coding system, for the following two reasons:

1. E provides more unique codes than N and K.
2. E gives a lower bound as good as N and K on a mean time versus variance graph.

After the robust parameter E has been determined, the experimental codes can be found using this parameter. To find the experimental codes, the graph, shown in Figure 2.3, was used. Each point in the graph represents a unique, modified variable length code. The dashed line in the graph emphasizes the lower bound which met the minimum variance criteria. The boxes on this line were picked as the best experimental codes, for a given mean and variance. Table 11 shows the respective mean times and variances of the experimental codes extracted from Figure 2.3. It can be also noticed that the other codes that do not appear in Table 11 are those that appear above the dashed line.

The codes in Table 11, are listed with their mean times in increasing order but their variances in decreasing order. Despite having the minimum average length, the Huffman code has the largest variance. On the other hand, code M has a variance close to zero but has the largest mean time. For

the given alphabet it is possible to obtain a variance of zero by using a block code. A block code gives an average length of 6 with zero variance. Finally the code words belonging to the various codes in Table 11, are given in Table 12.

We graph in Figure 2.4 only the experimental codes from Table 11, that have minimum variance for a given mean time. The extreme points, the Huffman code (code A) and the block code, also appear in this figure. This figure emphasizes that a small increase in average length can cause a large reduction in variance.

When the Huffman code is utilized as the reference for computing the increments in average lengths and the decrements in variances of these modified codes, the gain in variance versus the loss in mean time can be plotted as a difference from the reference Huffman code. This graph is given in Figure 2.5. The data for this figure appears in Table 13. The line segments between code M and the block code and between Huffman code and code B are almost parallel to the horizontal and vertical axes respectively. These parallel segments in Figure 2.5 show that, a little gain in one variable can result in a significant loss in the other variable. The last two columns in Table 13 give the relative gain and loss between adjacent experimental codes.

TABLE 8

MEAN TIMES AND VARIANCES OBTAINED BY USING N

N	MEAN TIME (L)	VARIANCE (V)
0	4.30771	1.918285
1	4.31277	1.416465
2	4.31439	1.457369
3	4.31940	1.344463
6	4.35005	1.221395
5	4.35935	0.939958
4	4.36186	0.937497
7	4.36270	1.019089
9	4.43946	0.590445
8	4.45705	0.529595
11	4.49867	0.501278
13	4.54577	0.472005
12	4.54593	0.472490
15	4.59967	0.433505
16	4.60637	0.434445
14	4.64103	0.366370
17	4.68298	0.421418
19	4.83615	0.292163
21	4.86950	0.331630
20	4.93958	0.204529
26	4.95533	0.220695
25	5.06011	0.057077
28	5.07212	0.067039
29	5.08843	0.080610

TABLE 9

MEAN TIMES AND VARIANCES OBTAINED BY USING K

K	MEAN TIME (L)	VARIANCE (V)
1.0	4.30771	1.918285
1.032	4.31055	1.414269
1.25	4.31077	1.414752
1.4	4.33760	1.213626
1.54	4.34335	1.127361
1.52	4.35746	0.933442
1.61	4.35969	0.935833
1.72	4.35996	0.927369
1.98	4.41490	0.672958
2.027	4.42819	0.593943
3.68	4.43733	0.448006
2.35	4.44997	0.531777
2.14	4.45058	0.533478
2.7	4.45617	0.527759
2.38	4.45790	0.527428
3.33	4.45856	0.530163
2.44	4.48506	0.484637
2.8	4.48745	0.486902
3.11	4.49203	0.480956
3.06	4.52360	0.440343
4.91	4.53024	0.446306
3.36	4.53074	0.442435
3.79	4.53755	0.446070
4.45	4.53823	0.446178
3.95	4.53861	0.447969
5.03	4.57248	0.404767
5.06	4.57824	0.409479
4.34	4.57902	0.401276
4.02	4.57940	0.403036
5.2	4.59420	0.416066
4.56	4.59470	0.415852
6.1	4.62853	0.362160
5.12	4.63942	0.363202
6.73	4.63986	0.363039
6.09	4.64008	0.369098
5.19	4.64025	0.365150
4.87	4.64075	0.364889
5.67	4.64729	0.370846
5.43	4.64768	0.372951
6.78	4.65639	0.381882
6.48	4.71585	0.319629
6.84	4.72401	0.328899

TABLE 9

MEAN TIMES AND VARIANCES OBTAINED BY USING K (cont'd.)

K	MEAN TIME (L)	VARIANCE (V)
7.48	4.73311	0.340340
9.76	4.79901	0.254473
9.75	4.80643	0.258361
7.59	4.80732	0.260114
16.2	4.89612	0.164169
9.95	4.90165	0.168137
12.11	4.90193	0.169472
9.29	4.90242	0.169338
13.006	4.90309	0.171358
12.12	4.90980	0.175424
10.39	4.91852	0.183301
15.96	5.02760	0.038182
14.84	5.02821	0.035494
12.79	5.03380	0.038178
16.21	5.04112	0.042269
14.58	5.04151	0.043287
13.45	5.05056	0.049824
19.0	5.06011	0.057077
53.22	5.07212	0.067039
272.84	5.08843	0.080610

TABLE 10

MEAN TIMES AND VARIANCES OBTAINED BY USING E

E	MEAN TIME (L)	VARIANCE (V)
0.0	4.30771	1.918285
0.00005	4.30781	1.917443
0.00011	4.30797	1.918384
0.00039	4.30803	1.921588
0.00034	4.30825	1.919612
0.00128	4.30836	1.923414
0.00045	4.30837	1.922558
0.00049	4.30858	1.922898
0.00055	4.30902	1.927887
0.00223	4.31025	1.931915
0.00183	4.31031	1.935338
0.00089	4.31059	1.932884
0.00139	4.31154	1.940723
0.00095	4.31181	1.745485
0.00447	4.31199	1.732892
0.00185	4.31265	1.730960
0.00106	4.31287	1.433402
0.00296	4.31293	1.932245
0.00441	4.31305	1.420810
0.00312	4.31337	1.929309
0.00357	4.31348	1.422090
0.00190	4.31371	1.420556
0.00230	4.31394	1.397182
0.02180	4.31438	1.469645
0.00425	4.31448	1.429542
0.00201	4.31476	1.392426
0.00274	4.31484	1.399356
0.00396	4.31583	1.432461
0.00218	4.31604	1.405119
0.00268	4.31694	1.407869
0.00419	4.31706	1.475493
0.00329	4.31727	1.446810
0.00502	4.31790	1.941310
0.00508	4.31828	1.721818
0.00687	4.31867	1.379099
0.00686	4.31895	1.389841
0.00496	4.31961	1.731779
0.00603	4.31986	1.362330
0.00658	4.32017	1.398981
0.00699	4.32046	1.343205
0.00607	4.32118	1.415403
0.00600	4.32217	1.425076
0.00626	4.32274	1.359079
0.00659	4.32275	1.748482
0.00625	4.32346	1.747414
0.00530	4.32381	1.359577
0.00643	4.32397	1.757613
0.00575	4.32431	1.342973
0.00503	4.32480	1.369245
0.00512	4.32497	1.366084

TABLE 10

MEAN TIMES AND VARIANCES OBTAINED BY USING E (cont'd.)

E	MEAN TIME (L)	VARIANCE (V)
0.00703	4.32526	1.356266
0.00703	4.32530	1.352640
0.00586	4.32547	1.349479
0.00542	4.32575	1.370917
0.01060	4.32654	1.368932
0.00453	4.32709	1.380862
0.00788	4.32759	1.364255
0.02330	4.33063	1.346434
0.00743	4.33118	1.359620
0.00871	4.33136	1.375061
0.00816	4.33145	1.358691
0.00715	4.33172	1.358022
0.00887	4.33185	1.377726
0.00877	4.33189	1.374099
0.02230	4.33246	1.354690
0.01140	4.33291	1.325861
0.01170	4.33329	1.334128
0.00771	4.33346	1.330964
0.00745	4.33357	1.366281
0.01130	4.33558	1.345706
0.01250	4.33631	1.235006
0.00978	4.34066	1.383511
0.00949	4.34194	1.393577
0.00866	4.34395	1.403128
0.01830	4.34613	1.358444
0.01660	4.34655	1.226893
0.01640	4.34710	1.221122
0.01760	4.34750	1.240114
0.01120	4.34771	1.233348
0.01750	4.34782	1.231681
0.01740	4.34867	1.334379
0.01400	4.35039	1.228117
0.01280	4.35066	1.227438
0.01370	4.35094	1.222601
0.01470	4.35251	1.335587
0.01460	4.35312	1.333486
0.01530	4.35323	1.346139
0.01540	4.35403	1.361652
0.01560	4.35409	1.365190
0.02190	4.36112	0.943632
0.01730	4.36404	1.242675
0.01500	4.36739	1.244890
0.01430	4.36766	1.243546
0.02050	4.36793	1.243258
0.02310	4.36810	1.157882
0.01710	4.36837	1.241394
0.01690	4.36898	1.239774
0.02390	4.36984	0.935738
0.06920	4.37127	0.957109
0.02000	4.37334	1.355217

TABLE 10

MEAN TIMES AND VARIANCES OBTAINED BY USING E (cont'd.)

E	MEAN TIME (L)	VARIANCE (V)
0.02250	4.37395	1.353091
0.04920	4.38193	0.921279
0.02370	4.38279	1.147962
0.03910	4.38336	0.880996
0.04230	4.38370	0.893874
0.03960	4.38381	0.892199
0.05220	4.38416	0.902081
0.02270	4.38462	1.152367
0.01920	4.38508	0.934533
0.02080	4.38514	0.937947
0.02090	4.38538	0.948022
0.05170	4.38551	0.972232
0.04140	4.38553	0.897277
0.01950	4.38565	0.947064
0.02160	4.38579	1.388296
0.01860	4.38592	0.946366
0.02100	4.38933	0.936852
0.02060	4.38978	1.268812
0.03060	4.39034	0.951535
0.03120	4.39147	0.967141
0.03940	4.39335	0.907766
0.02510	4.39431	1.375608
0.02440	4.39480	1.377233
0.02970	4.39537	1.389753
0.02930	4.39564	1.389049
0.03340	4.39576	0.858914
0.03390	4.39600	0.868984
0.02430	4.39608	1.387160
0.02590	4.39627	0.868020
0.02460	4.39669	1.385007
0.02600	4.39699	0.878509
0.03430	4.39852	0.898482
0.03280	4.39995	0.857730
0.02950	4.40000	1.390720
0.02920	4.40033	0.891206
0.02570	4.40096	0.872391
0.03220	4.40111	0.861741
0.02580	4.40199	0.889354
0.03360	4.40209	0.887974
0.03310	4.40308	0.897487
0.02020	4.40518	1.173649
0.02540	4.40654	0.956005
0.02420	4.40753	0.963529
0.02530	4.40770	0.960007
0.02520	4.40982	0.971088
0.03270	4.41251	0.901085
0.02860	4.41647	0.947963
0.03860	4.41956	0.814329
0.05140	4.42000	0.812420
0.05150	4.42236	0.849492

TABLE 10

MEAN TIMES AND VARIANCES OBTAINED BY USING E (cont'd.)

E	MEAN TIME (L)	VARIANCE (V)
0.05920	4.42297	0.804586
0.03890	4.42520	0.825645
0.03080	4.42794	0.886547
0.03050	4.42893	0.894029
0.05900	4.43553	0.847364
0.02720	4.43861	0.812091
0.02790	4.43889	0.806646
0.02700	4.43916	0.805118
0.05730	4.43943	0.803851
0.02820	4.43956	0.823267
0.03770	4.43960	0.819552
0.02680	4.43977	0.816012
0.02810	4.43988	0.814106
0.02870	4.44159	0.852228
0.03660	4.44186	0.851240
0.05490	4.44680	0.829550
0.03160	4.44847	0.955625
0.06390	4.45343	0.541571
0.03030	4.45502	0.826637
0.03040	4.45513	0.839047
0.04110	4.46079	0.771663
0.04080	4.46140	0.774930
0.04200	4.46315	0.814042
0.06260	4.46843	0.594983
0.06400	4.46935	0.542661
0.06370	4.46962	0.541117
0.06200	4.46989	0.539833
0.06320	4.47023	0.551974
0.04990	4.47085	0.757790
0.04960	4.47140	0.750782
0.03510	4.47167	0.749497
0.03600	4.47180	0.768905
0.03540	4.47184	0.765187
0.04260	4.47201	0.761637
0.03890	4.47384	0.762056
0.03780	4.47407	0.864048
0.03620	4.47410	0.795983
0.03580	4.47437	0.795983
0.02650	4.47451	0.861210
0.02620	4.47512	0.857741
0.06410	4.47598	0.557923
0.06480	4.47604	0.561206
0.03530	4.47776	0.767565
0.04810	4.47904	0.774721
0.05360	4.48048	0.505279
0.08530	4.48150	0.604438
0.08490	4.48231	0.509327
0.08310	4.48583	0.507199
0.06930	4.48795	0.517055
0.10600	4.48847	0.595987

TABLE 10

MEAN TIMES AND VARIANCES OBTAINED BY USING E (cont'd.)

E	MEAN TIME (L)	VARIANCE (V)
0.06620	4.48988	0.610038
0.03460	4.49166	0.819630
0.05840	4.49231	0.484761
0.05790	4.49258	0.483465
0.08130	4.49292	0.495590
0.05820	4.49867	0.501278
0.06080	4.49475	0.495932
0.07090	4.49830	0.562617
0.07950	4.49995	0.508380
0.06570	4.50765	0.676581
0.06730	4.51257	0.552762
0.05060	4.53120	0.728027
0.05070	4.53147	0.726449
0.05950	4.56080	0.477003
0.06000	4.56179	0.482242
0.05960	4.56196	0.478320
0.07910	4.56798	0.451079
0.07690	4.56859	0.415855
0.09190	4.56914	0.408740
0.09510	4.56920	0.457171
0.07670	4.56941	0.407402
0.09060	4.56958	0.423059
0.07020	4.56975	0.419475
0.09050	4.56981	0.454807
0.07350	4.57082	0.436364
0.08720	4.57158	0.419536
0.09010	4.57550	0.424280
0.09070	4.57556	0.427551
0.11000	4.57678	0.431185
0.10300	4.57818	0.406448
0.05870	4.57869	0.414348
0.07600	4.57896	0.413005
0.09200	4.57902	0.401276
0.07770	4.57940	0.410076
0.08670	4.58001	0.406478
0.07440	4.59420	0.416067
0.09290	4.59527	0.430664
0.07360	4.60542	0.455207
0.12600	4.62106	0.457004
0.07870	4.64025	0.365150
0.20000	4.68298	0.421418
0.13100	4.69883	0.323067
0.19900	4.70024	0.446184
0.12700	4.70066	0.322656
0.10400	4.70418	0.311951
0.13300	4.73267	0.342165
0.13200	4.73389	0.342975
0.17500	4.89183	0.188969
0.17100	4.89779	0.168143
0.19100	5.04151	0.043287
0.26400	5.08843	0.080610

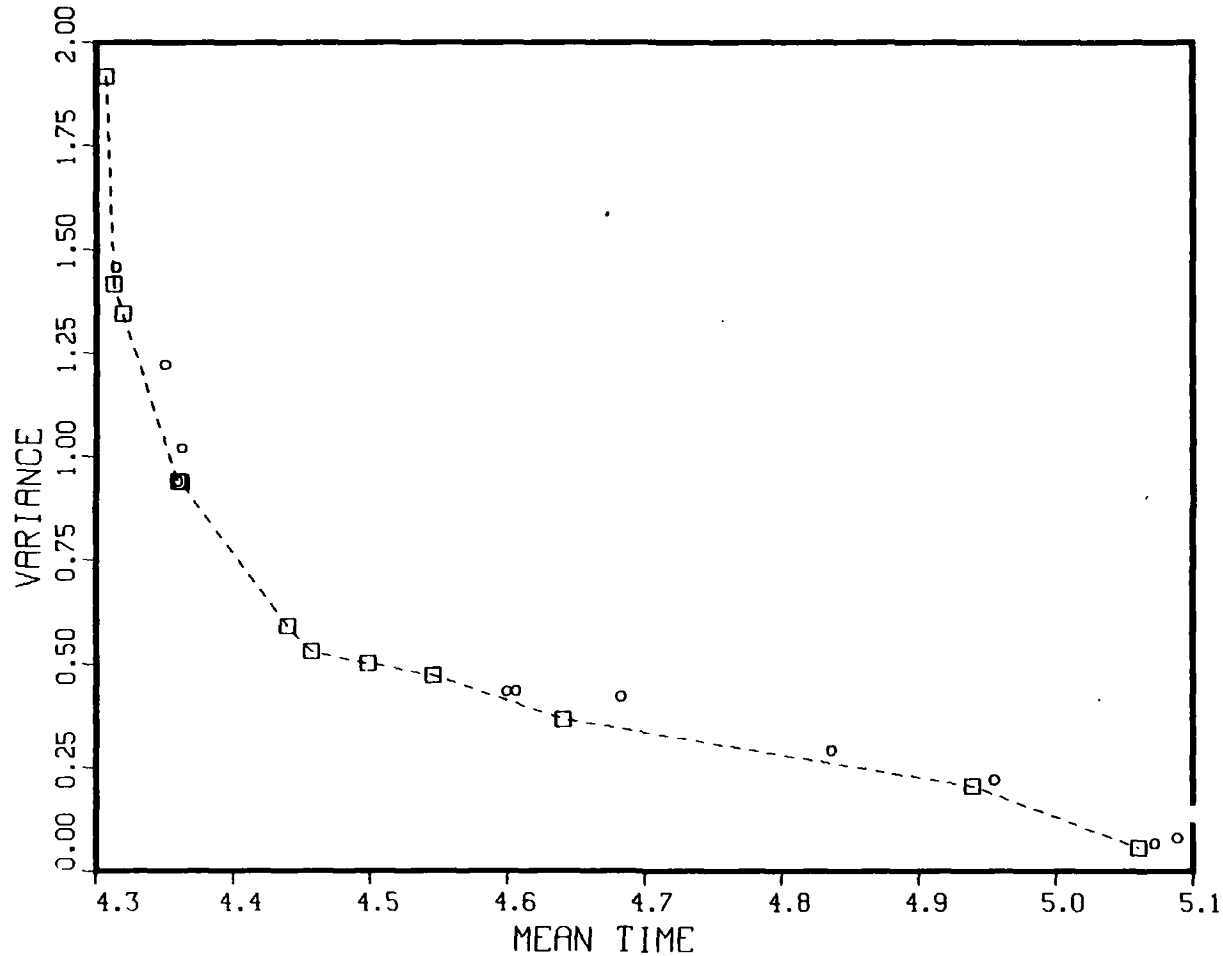


Figure 2.1 Mean time - Variance Trade-off for the Parameter N.

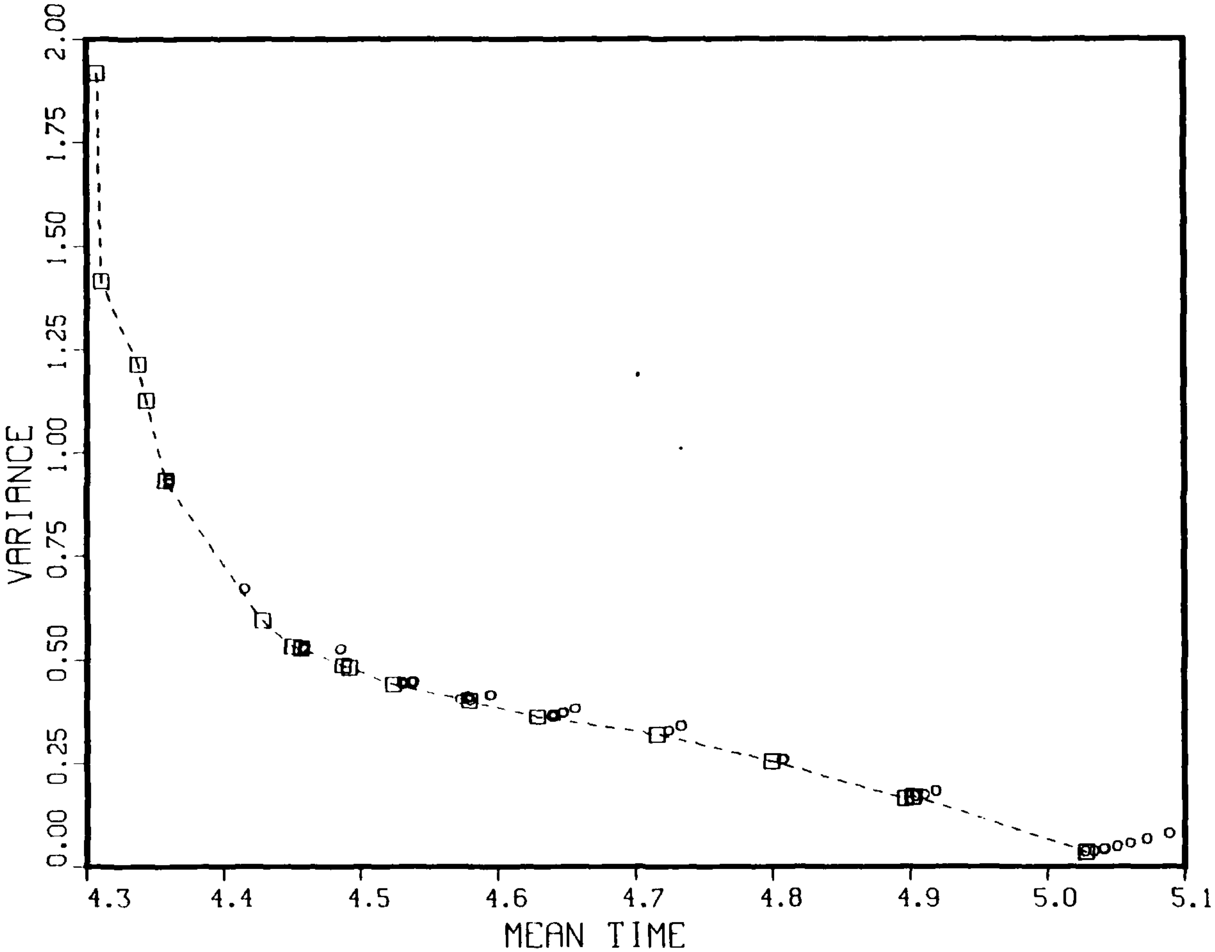


Figure 2.2 Mean time - Variance Trade-off for the Parameter K.

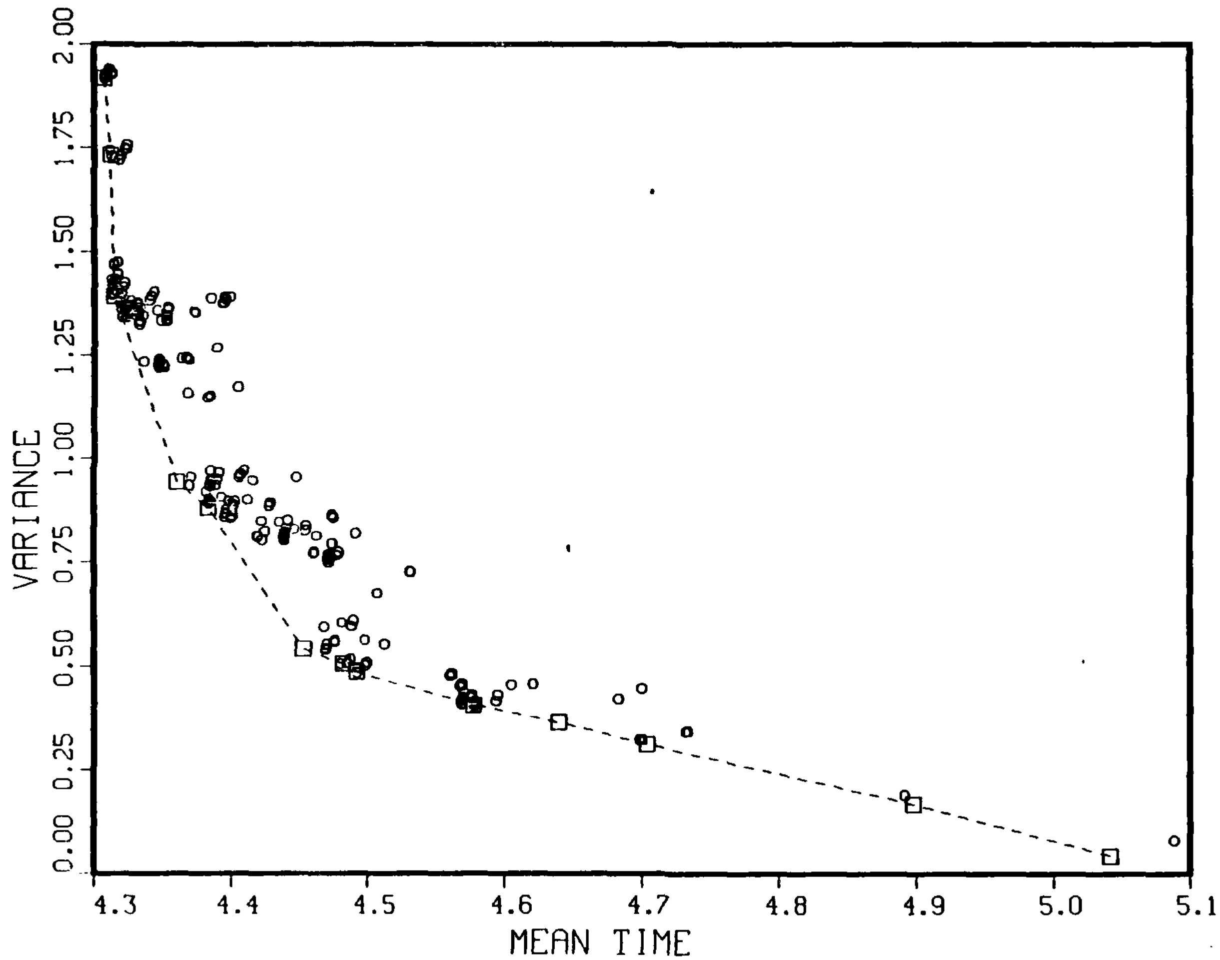


Figure 2.3 **Mean time - Variance Trade-off for the Parameter E.**

TABLE 11

MEAN TIMES AND VARIANCES OF THE EXPERIMENTAL CODES

CODE NAME	MEAN TIME	VARIANCE
A (HUFFMAN CODE)	4.30771	1.918285
B	4.31199	1.732892
C	4.31476	1.392426
D	4.36112	0.943632
E	4.38336	0.880996
F	4.45343	0.541571
G	4.48231	0.509327
H	4.49231	0.484761
I	4.57818	0.406448
J	4.64025	0.365150
K	4.70418	0.311951
L	4.89779	0.168143
M	5.04151	0.043287

TABLE 12

MODIFIED HUFFMAN CODES

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	010	F	10010011
I	101	0	100100100
A	111	'	100101101
E	0001	1	0000011100
N	0110	"	0000011101
R	1000	2	1001001011
U	1100	)	1001011001
L	1101	5	1001011000
S	00100	3	1001011110
K	00101	8	1001011101
D	00111	(	1001011111
T	01110	4	00000111010
M	01111	;	00000111011
Y	10011	9	00000111101
O	000000	J	10010010101
G	000010	6	10010010100
B	001100	W	10010111000
C	001101	:	10010111001
,	0000010	7	000001111001
.	0000110	-	0000011110000
Z	0000111	?	00000111100011
V	1001000	X	000001111000100
P	1001010	Q	000001111000101
H	00000110		

CODE NAME : A
(HUFFMAN CODE)

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	011	F	11001000
I	101	0	11001001
A	111	'	000101101
E	0010	1	000101110
N	0101	"	000101111
R	1000	2	000010000
U	1101	)	000010001
L	00000	5	0001010100
S	00011	3	0001010110
K	00111	8	0001010101
D	01000	(	0001010111
T	01001	4	0001011000
M	10010	;	0000011001
Y	10011	9	00010100010
O	11000	J	00010100011
G	000100	6	00010100100
B	001100	W	00010100110
C	001101	:	00010100101
,	110011	7	00010100111
.	0000101	-	000101000000
Z	0000110	?	000101000001
V	0000111	X	000101000010
P	1100101	Q	000101000011
H	00001001		

CODE NAME : B

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	010	F	01111110
I	101	0	011111000
A	0000	'	011111001
E	0001	1	011111010
N	0110	"	011111011
R	1000	2	111110100
U	1101	)	111110110
L	1110	5	111110101
S	00100	3	111110111
K	00110	8	1100101010
D	00111	(	1100101100
T	01110	4	1100101011
M	10010	;	1100101101
Y	10011	9	1100101110
O	11000	J	1100101111
G	11110	6	0111111100
B	001011	W	0111111101
C	011110	:	0111111110
,	110011	7	0111111111
.	111111	-	11001010000
Z	0010100	?	11001010010
V	0010101	X	11001010001
P	1100100	Q	11001010011
H	1111100		

CODE NAME : C

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	010	F	0111111
I	0000	0	11011000
A	0001	'	11011001
E	0011	1	11011010
N	1010	"	11011100
R	1100	2	11011011
U	1110	)	11011110
L	1111	5	11011101
S	01110	3	11011111
K	10000	8	011110110
D	00100	(	011110111
T	00101	4	0111100000
M	11010	;	0111100001
Y	10010	9	0111100010
O	10011	J	0111100100
G	10110	6	0111100011
B	10111	W	0111100101
C	011000	:	0111100110
,	011001	7	0111100111
.	011010	-	0111101000
Z	011011	?	0111101010
V	100010	X	0111101001
P	100011	Q	0111101011
H	0111110		

CODE NAME : D

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	101	F	0010011
I	0001	0	00100000
A	0101	'	00100001
E	0111	1	00100010
N	1000	"	00100011
R	1001	2	001110000
U	1111	)	001110001
L	00110	5	001110010
S	11001	3	001110100
K	00000	8	001110011
D	00001	(	001110110
T	01000	4	001110101
M	01001	;	001111000
Y	11010	9	001110111
O	11011	J	001111001
G	01100	6	001111010
B	01101	W	001111011
C	11100	:	001111110
,	11101	7	001111111
.	001010	-	0011111000
Z	001011	?	0011111010
V	110000	X	0011111001
P	110001	Q	0011111011
H	0010010		

CODE NAME : E

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	0001	F	0010011
I	0110	0	1001100
A	1000	'	1001101
E	1011	1	1001110
N	1100	"	1001111
R	1101	2	10010100
U	1110	)	10010110
L	1111	5	10010101
S	00111	3	10010111
K	00000	8	00100000
D	00001	(	00100001
T	01000	4	00100010
M	01001	;	00100011
Y	01010	9	100100010
O	01011	J	100100011
G	01110	6	100100100
B	01111	W	100100101
C	10100	:	100100110
,	10101	7	100100111
.	001010	-	1001000000
Z	001011	?	1001000010
V	001100	X	1001000001
P	001101	Q	1001000011
H	0010010		

CODE NAME : F

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	0100	F	010111
I	1000	0	10010110
A	1011	'	10010111
E	1100	1	10011000
N	1101	"	10011001
R	1110	2	10011010
U	1111	)	10011100
L	00000	5	10011011
S	00001	3	10011110
K	00010	8	10011101
D	00011	(	10011111
T	00100	4	100100000
M	00101	;	100100001
Y	00110	9	100100010
O	00111	J	100100100
G	01100	6	100100011
B	01101	W	100100101
C	01110	:	100100110
,	01111	7	100100111
.	101000	-	100101000
Z	101011	?	100101010
V	010100	X	100101001
P	010101	Q	100101011
H	010110		

CODE NAME : G

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	0000	F	001101
I	0111	0	0001000
A	1001	'	0001001
E	1010	1	0001010
N	1101	"	0001011
R	1110	2	1011110
U	1111	)	1011111
L	00101	5	10110100
S	00111	3	10110110
K	01000	8	10110101
D	01001	(	10111000
T	01010	4	10110111
M	01011	;	10111010
Y	00110	9	10111001
O	01101	J	10111011
G	10000	6	101100000
B	10001	W	101100010
C	11000	:	101100001
,	11001	7	101100011
.	000110	-	101100100
Z	000111	?	101100101
V	001000	X	101100110
P	001001	Q	101100111
H	001100		

CODE NAME : H

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	1000	F	001101
I	1011	0	001110
A	1101	'	001111
E	1110	1	1001100
N	1111	"	1001101
R	00000	2	1001110
U	00001	)	1001111
L	00010	5	0010000
S	00011	3	0010010
K	01000	8	0010001
D	01001	(	0010011
T	01010	4	10010100
M	01011	;	10010110
Y	01100	9	10010101
O	01101	J	10010111
G	01110	6	100100000
B	01111	W	100100010
C	10100	:	100100001
,	10101	7	100100011
.	11000	-	100100100
Z	11001	?	100100101
V	001010	X	100100010
P	001011	Q	100100111
H	001100		

CODE NAME : I

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	0101	F	011011
I	1010	0	000000
A	1100	'	000001
E	1110	1	1011100
N	00001	"	1011101
R	00010	2	1011110
U	00011	)	1011111
L	00100	5	1101000
S	00101	3	1101001
K	00110	8	1101010
D	00111	(	1101100
T	01000	4	1101011
M	01001	;	1101110
Y	01110	9	1101101
O	01111	J	1101111
G	10000	6	10110000
B	10001	W	10110001
C	10010	:	10110010
,	10011	7	10110100
.	11110	-	10110011
Z	11111	?	10110101
V	011000	X	10110110
P	011001	Q	10110111
H	011010		

CODE NAME : J

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	1011	F	000001
I	1100	0	000010
A	1101	'	000011
E	00010	1	1110100
N	00011	"	1110101
R	00100	2	1110110
U	00101	)	1110111
L	00110	5	1111000
S	00111	3	1111001
K	01000	8	1111010
D	01001	(	1111011
T	01010	4	1111100
M	01011	;	1111110
Y	01100	9	1111101
O	01101	J	1111111
G	01110	6	11100000
B	01111	W	11100010
C	10000	:	11100001
,	10001	7	11100011
.	10010	-	11100100
Z	10011	?	11100110
V	10100	X	11100101
P	10101	Q	11100111
H	000000		

CODE NAME : K

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	1111	F	010010
I	00000	0	010001
A	00001	'	010011
E	00010	1	010100
N	00011	"	010110
R	01100	2	010101
U	01101	)	010111
L	01110	5	0010000
S	01111	3	0010001
K	10000	8	0010010
D	10001	(	0010011
T	10010	4	0010100
M	10011	;	0010101
Y	10100	9	0010110
O	10101	J	0010111
G	10110	6	0011000
B	10111	W	0011001
C	11000	:	0011010
,	11001	7	0011011
.	11010	-	0011100
Z	11100	?	0011110
V	11011	X	0011101
P	11101	Q	0011111
H	010000		

CODE NAME : L

TABLE 12

MODIFIED HUFFMAN CODES (cont'd.)

SYMBOL	CODE WORDS	SYMBOL	CODE WORDS
space	00011	F	000011
I	00100	0	100000
A	00101	'	100001
E	01010	1	100010
N	01011	"	100011
R	01100	2	100100
U	01101	)	100110
L	01110	5	100101
S	01111	3	100111
K	10100	8	010000
D	10101	(	010001
T	10110	4	010010
M	10111	;	010011
Y	11000	9	000100
O	11001	J	000101
G	11010	6	0011000
B	11011	W	0011010
C	11100	:	0011001
,	11101	7	0011011
.	11110	-	0011100
Z	11111	?	0011110
V	000000	X	0011101
P	000001	Q	0011111
H	000010		

CODE NAME : M

TABLE 13

GAIN AND LOSS OF THE EXPERIMENTAL CODES

CODE	GAIN IN VARIANCE	LOSS IN MEAN TIME	RELATIVE GAIN IN VARIANCE	RELATIVE LOSS IN MEAN TIME
A	0	0	--	--
B	0.18539	0.00428	0.18539	0.00428
C	0.52586	0.00705	0.34047	0.00277
D	0.97465	0.05341	0.44879	0.04636
E	1.03729	0.07565	0.06264	0.02224
F	1.37671	0.14572	0.33932	0.07007
G	1.41296	0.17460	0.03624	0.02888
H	1.43352	0.18460	0.02057	0.01
I	1.51184	0.27047	0.07831	0.08587
J	1.55314	0.33254	0.04130	0.06207
K	1.60633	0.39647	0.05320	0.06393
L	1.75014	0.59008	0.14381	0.19361
M	1.87500	0.73380	0.12486	0.14372
BLOCK	1.91829	1.69229	0.04329	0.95849

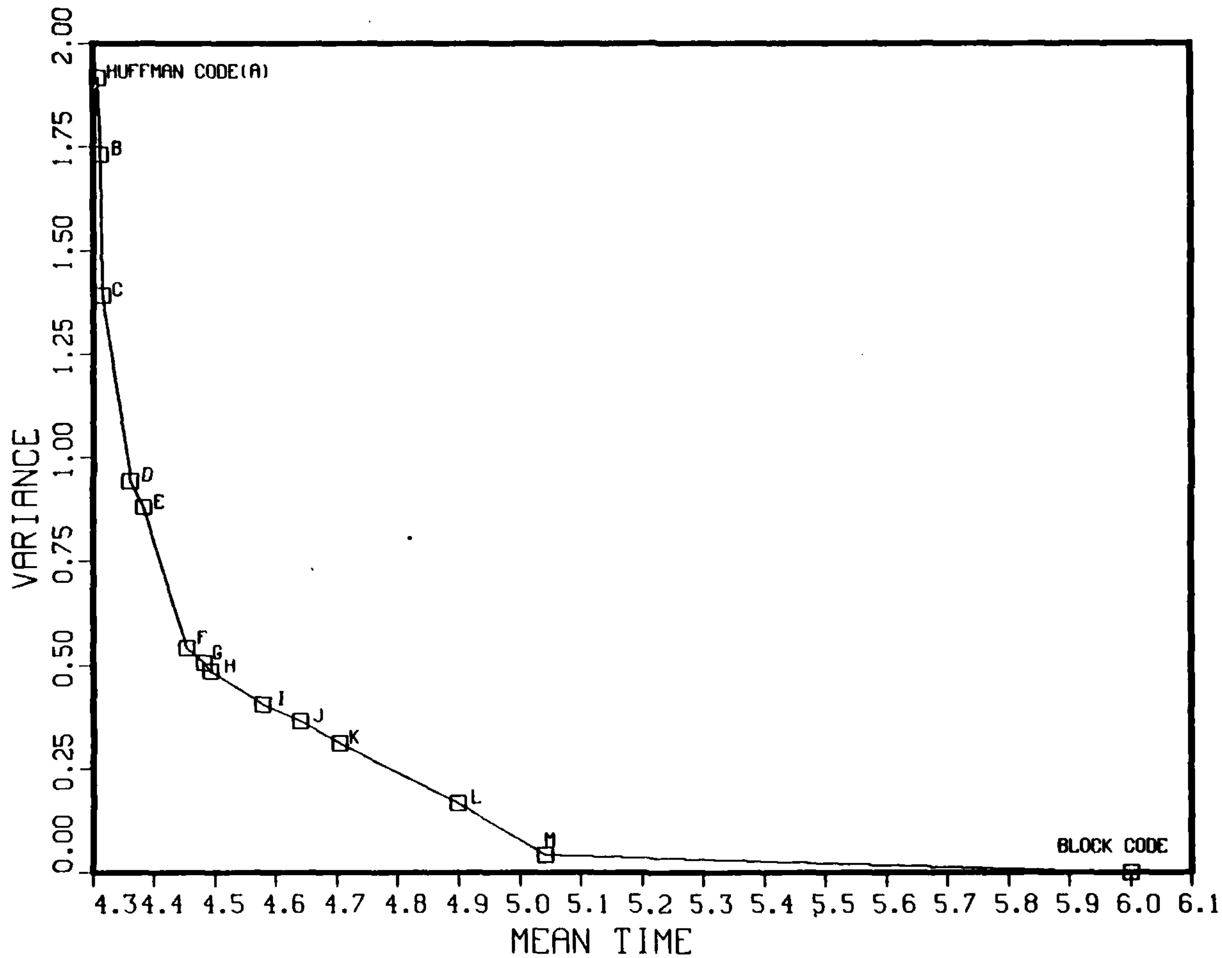


Figure 2.4 Mean time - Variance Trade-off for Experimental Codes.

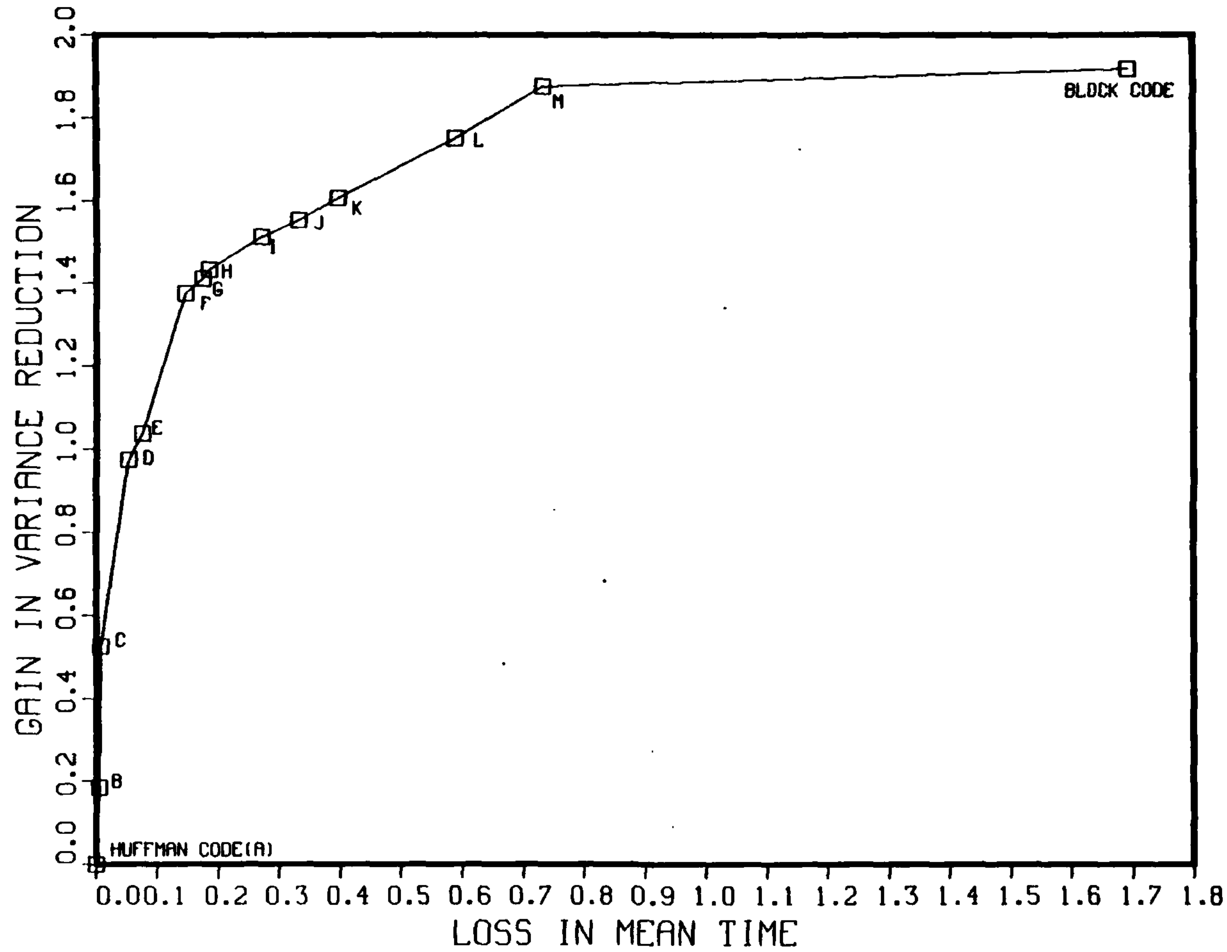


Figure 2.5 Gain and Loss in Experimental Codes.

III. REDUCTION IN BANDWIDTH

A. BACKGROUND ON QUEUEING THEORY

As far as on-line communication is concerned, information can be sent through the channel either by storing it on a device such as a tape and forwarding it later, or by transmitting it immediately.

The flow through the communication channel is defined by an arrival pattern of messages which come to the channel to be communicated during a certain interval of time. In order to be able to satisfy the demands placed on the network, the channel capacity of the channel should be sufficient to handle the average rate of flow. For a single server (channel) the relationship between the input and output rate is defined as $R \leq C$ [Ref. 9], where R represents the average arrival rate of the source symbols to the system (input rate) and C stands for the capacity of a communication processor (transmission rate) for handling the traffic.

The input rate can be made equal to the output rate ($R=C$) only when a steady flow occurs. Steady flow means, that after encoding the source letters as a block code so that the same number of digits belong to each symbol, the symbols that arrive at the processor at each unit time, are accepted, by the channel at the same rate that they arrive. In this way there is no need to have a waiting line or buffer since at each unit of time the arriving digits (0 or 1) can be sent immediately over the channel. On the other hand, when fluctuations or unsteady flows exist in the channel, even with $R<C$ condition, a waiting line can build up and the processor must put the excess digits in a buffering device. These excess digits, stored in the buffer, are later forwarded on a first in first out (FIFO) basis to accomplish the transmission. If the buffer becomes full, the arriving digits will be lost and overflow occurs. Therefore,

the size of the buffer selected in order to prevent overflow, should be large enough to enable transmission of the entire messages through the single channel. Overflow also will occur when R>C. The waiting line grows without bound and the system overflows.

B. TRANSMISSION OF FINITE LENGTH MESSAGES

As was explained in the previous chapters, both the Huffman code itself and the modified Huffman codes are variable length codes. During transmission of a coded message, which is written by using the particular alphabet given in Table 6, the system forms an unsteady flow into the communication network. This occurs since encoded letters generally consist of different lengths of digits. Therefore, according to the discussion in the previous section about queueing theory, it is obvious that we will require a finite length buffer when a finite length message is transmitted. This is true no matter which of the codes of Table 11 are used. In Table 11 the mean times for each code actually represents the average input rates.

As a first step, the first 100 characters of the first magazine article given in Appendix A, were transmitted at different input and output rates in order to observe the variations of the maximum number of digits appearing in the buffer. To simulate the transmission, a computer program in the Fortran programming language was used. This Fortran language program which was written by the author, appears in Appendix C. The results of the simulation are shown in Table 14.

The first column of Table 14 stands for the input rates of the experimental codes from A to M. The block code was also included in these codes for comparison purposes. To be able to transmit a message using this particular alphabet, with the block code, each letter will consist of 6 digits as mentioned before and the output rate should also be a minimum of 6 bits per unit time, in order to handle the

TABLE 14

MAXIMUM BUFFERS WITH VARIOUS INPUT AND OUTPUT RATES

	28.2%	25%	20%	15%	10%	5%	BLOCK
OUT.RATES	4.30771	4.5	4.8	5.1	5.4	5.7	6.0

IN.RATES							
4.30771	42	35	25	23	21	19	17
4.31199	34	27	21	19	17	15	13
4.31476	36	29	23	21	19	17	15
4.36112	29	22	18	16	14	12	10
4.38336	30	23	19	17	15	13	11
4.45343	30	20	14	12	10	8	6
4.48231	38	26	17	15	13	11	9
4.49231	33	21	14	12	10	8	6
4.57818	36	19	8	6	4	2	1
4.64025	44	24	10	6	4	2	1
4.70418	46	26	10	6	4	2	1
4.89779	65	45	16	6	4	2	1
5.04151	80	60	30	5	3	1	0
6.00000	170	150	120	90	60	30	0

traffic without an overflow. Hence, whenever a coded symbol arrives with 6 bits per unit time at the processor, it will be accepted at the same rate by the channel and there will be no need for a buffer. For this reason, 6 bits per unit time both for the input and output rate was chosen as a basis for comparison of performance.

To show how much can be saved in the channel capacity by using variable length codes instead of block codes, the selected output rates of 5.7, 5.4, 5.1, 4.8, 4.5 and 4.30771 (Huffman code rate) bits per unit time are used, which are in fact 5%, 10%, 15%, 20%, 25% and 28.2% less than the block

code output rate, respectively. Savings higher than 28.2% was not considered according to the R and C relationship. No code can have a lower rate than the Huffman code. Then, for each input rate belonging to one of the experimental codes, the maximum sizes of the buffers were given for the corresponding 7 different output rates. As shown in Table 14, for the input and output rate of 6 bits per unit time for the block code, the maximum buffer size is 0.

Figure 3.1 illustrates various curves for the maximum buffer lengths versus the different input rates for the experimental codes. The output rate was kept the same for all of these input rates. Table 14 was used as the data for these curves. This figure clearly displays that there is a drop in the buffer lengths when the output rate approaches 6 bits per unit time for each unique code from A to M and also for the block code.

Three different codes, the Huffman code, block code, and code F were chosen by the author to observe the effects of variance on the buffer lengths. The number of digits in the buffer for each symbol in the 100 character message was obtained by running the same program given in Appendix C. The results are shown in Table 15. The output rate for these three different codes was selected to be the same, equal to 6 bits per unit time.

The curves shown in Figure 3.2 were plotted by using the data given in Table 15. The horizontal axis contains each character from 1 to 100. The corresponding buffer lengths are placed in the vertical axis. Figure 3.2 illustrates that the change of the buffer sizes and the required maximum buffer lengths for the Huffman code is much larger than the other two coding schemes due to the Huffman code's large variance. On the other hand, code F has a variance between the Huffman code and the block code. Therefore, the variability of the buffer lengths is less than that of the Huffman code, but it is more than that of the block code.

The dashed line represents the curve for code F. The block code needs no buffer with this chosen output rate. Therefore, the plot belonging to the block code is a straight line. For convenience, to distinguish the plots from each other, they were moved to their input rates level. Then, for example, when the Huffman code needs to have a buffer length of 1 at the 21st character, its curve jumps from 4.30771 to 5.30771 and whenever there is no need for a buffer the curve remains at the 4.30771 level.

C. TRANSMISSION OF THE LONG MESSAGES

The use of the smaller capacity than the block code output rate causes a reduction in the bandwidth demands. As far as base performance is concerned, a 25% reduction in capacity means also a 25% saving in the bandwidth. Note in Table 14, a saving of 28.2%, greater than the 25% savings, is discussed. The Huffman code rate of 4.30771 which gives 28.2% reduction requires larger buffers. Except for the Huffman code, since it was used to send more than this rate can handle, the buffer sizes required continue to grow as the length of the messages increase. For this reason the bandwidth which saves 25% was determined as a best output rate (4.5 bits per unit time). The different buffer lengths for different lengths of messages are given in Table 16. The message lengths were arbitrarily selected by the author to also include the entire two magazine articles. The output rate was held fixed at 4.5 bits per unit time and the buffer lengths required were obtained by using the program in Appendix C. Table 16 shows the fact that when the input rates become larger than the output rate, the buffer sizes increase with longer messages.

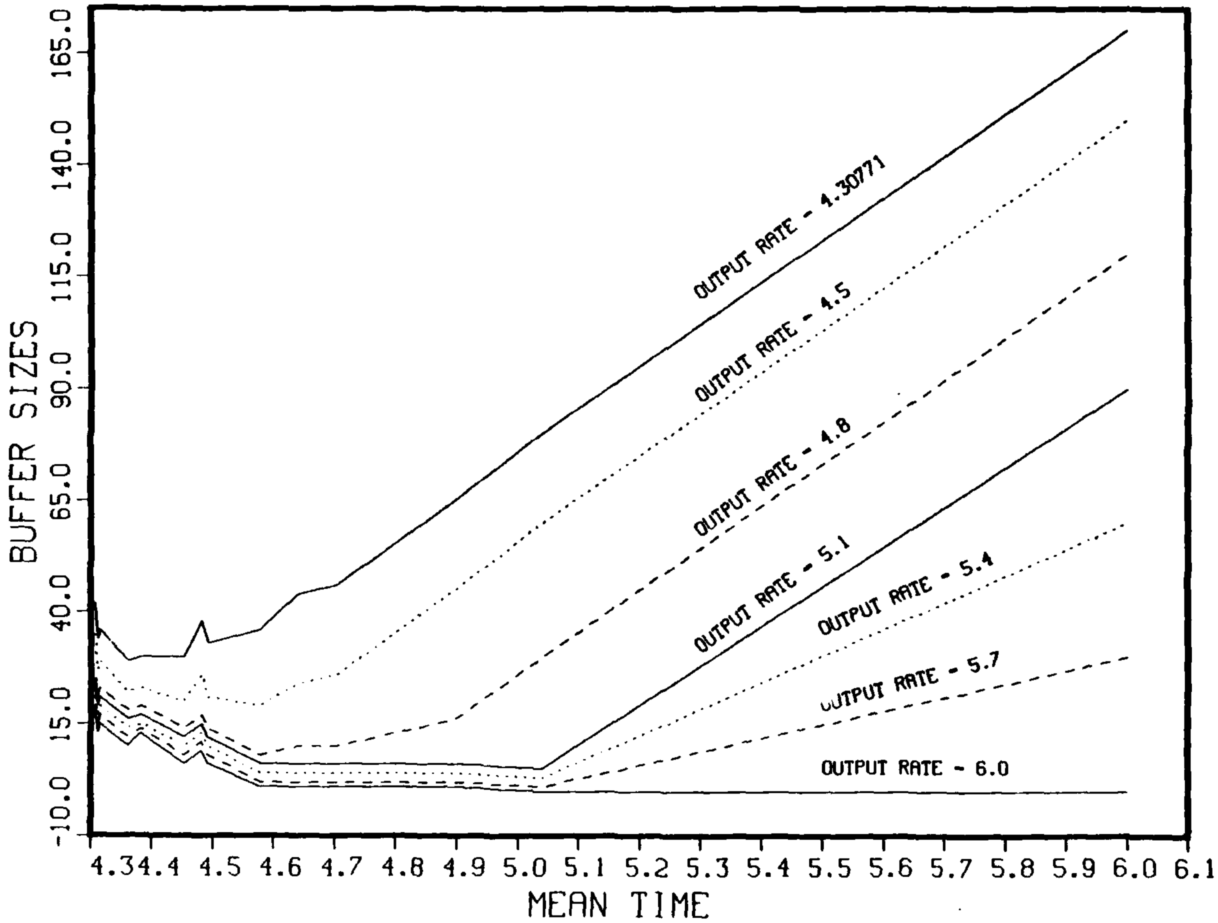


Figure 3.1 Maximum Buffer Sizes with Different Input and Output Rates.

TABLE 15

OBSERVED BUFFER LENGTHS FOR THE FIRST 100 CHARACTERS

CHARACTERS	HUFFMAN CODE	BLOCK CODE	CODE F
	BUFFER LENGTHS		
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	1	0	0
22	0	0	0
23	0	0	0
24	0	0	0
25	0	0	0
26	0	0	0
27	0	0	0
28	0	0	0
29	0	0	0
30	0	0	0
31	0	0	0
32	1	0	0
33	0	0	0
34	0	0	0
35	0	0	0
36	0	0	0
37	0	0	0
38	0	0	0
39	0	0	0
40	0	0	0
41	0	0	0
42	0	0	0
43	0	0	0
44	1	0	0
45	0	0	0
46	0	0	0
47	0	0	0
48	0	0	0
49	0	0	0
50	0	0	0

OUTPUT RATE : 6 BITS PER UNIT TIME

TABLE 15

OBSERVED BUFFER LENGTHS FOR THE FIRST 100 CHARACTERS (cont'd)

CHARACTERS	HUFFMAN CODE	BLOCK CODE	CODE F
	BUFFER LENGTHS		
51	0	0	0
52	0	0	0
53	0	0	0
54	0	0	0
55	0	0	0
56	0	0	0
57	0	0	0
58	0	0	0
59	0	0	0
60	0	0	0
61	0	0	0
62	0	0	0
63	0	0	0
64	0	0	0
65	0	0	0
66	1	0	0
67	0	0	0
68	4	0	2
69	7	0	3
70	8	0	3
71	11	0	4
72	14	0	5
73	17	0	6
74	14	0	4
75	13	0	3
76	13	0	2
77	11	0	0
78	9	0	0
79	7	0	0
80	6	0	0
81	3	0	0
82	3	0	0
83	0	0	0
84	0	0	0
85	0	0	0
86	4	0	1
87	4	0	0
88	3	0	0
89	2	0	0
90	0	0	0
91	0	0	0
92	0	0	0
93	0	0	0
94	1	0	0
95	0	0	0
96	0	0	0
97	1	0	0
98	0	0	0
99	4	0	1
100	2	0	0

OUTPUT RATE : 6 BITS PER UNIT TIME

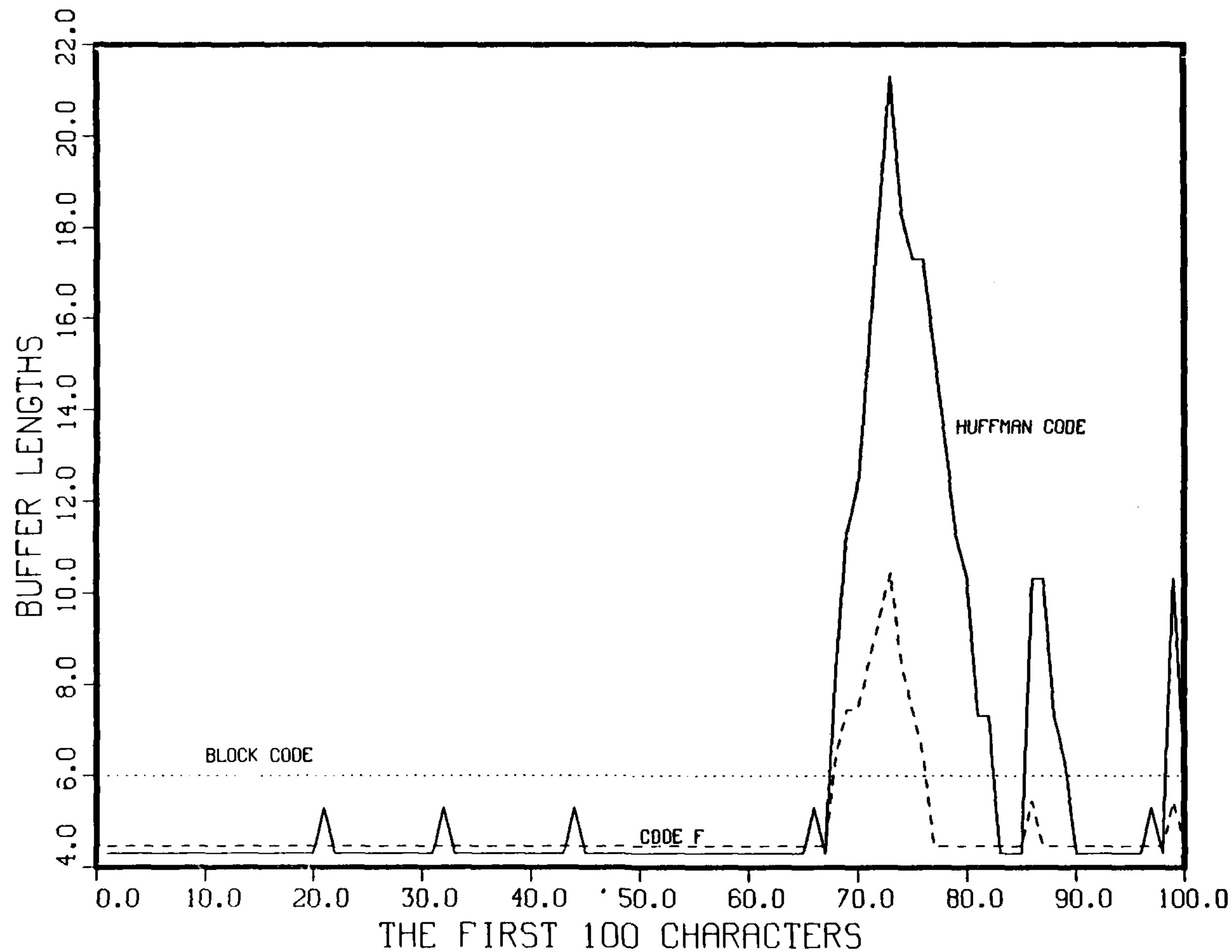


Figure 3.2 **Change of Buffer Lengths for the First 100 Characters.**

TABLE 16

MAXIMUM BUFFERS FOR DIFFERENT MESSAGE LENGTHS

NUMBER OF CHARACTERS	5000	10000	12000	15000	ENTIRE ARTICLES
INPUT RATES	MAXIMUM	BUFFER	LENGTHS		
4.30771 (Huffman)	35	49	63	63	63
4.31199 (code B)	27	44	56	56	56
4.31476 (code C)	29	43	47	47	47
4.36112 (code D)	22	34	41	41	41
4.38336 (code E)	23	36	43	43	43
4.45343 (code F)	20	39	59	59	59
4.48231 (code G)	28	77	113	117	117
4.49231 (code H)	22	93	129	150	150
4.57818 (code I)	316	723	856	1163	1376
4.64025 (code J)	644	138	1643	2122	2467
4.70418 (code K)	999	2063	2448	3087	3581
4.89779 (code L)	1965	3976	4765	5971	7010
5.04151 (code M)	2657	5417	6486	8120	9556
6.00000 (Block)	7500	15000	18000	22500	26480

OUTPUT RATE : 4.5 BITS PER UNIT TIME

From code F to the block code the maximum buffer lengths increase proportionally by increasing the message length. During the transmission of the two magazine articles a graph of the maximum buffer length versus the mean time is illustrated in Figure 3.3. The last column of Table 16 used as the data in this graph. Figure 3.3 illustrates that code D requires the minimum buffer size among all of the experimental codes. Although the Huffman code produces the minimum average length code, because of its large variance, it causes more delay at some part of the transmission than code D.

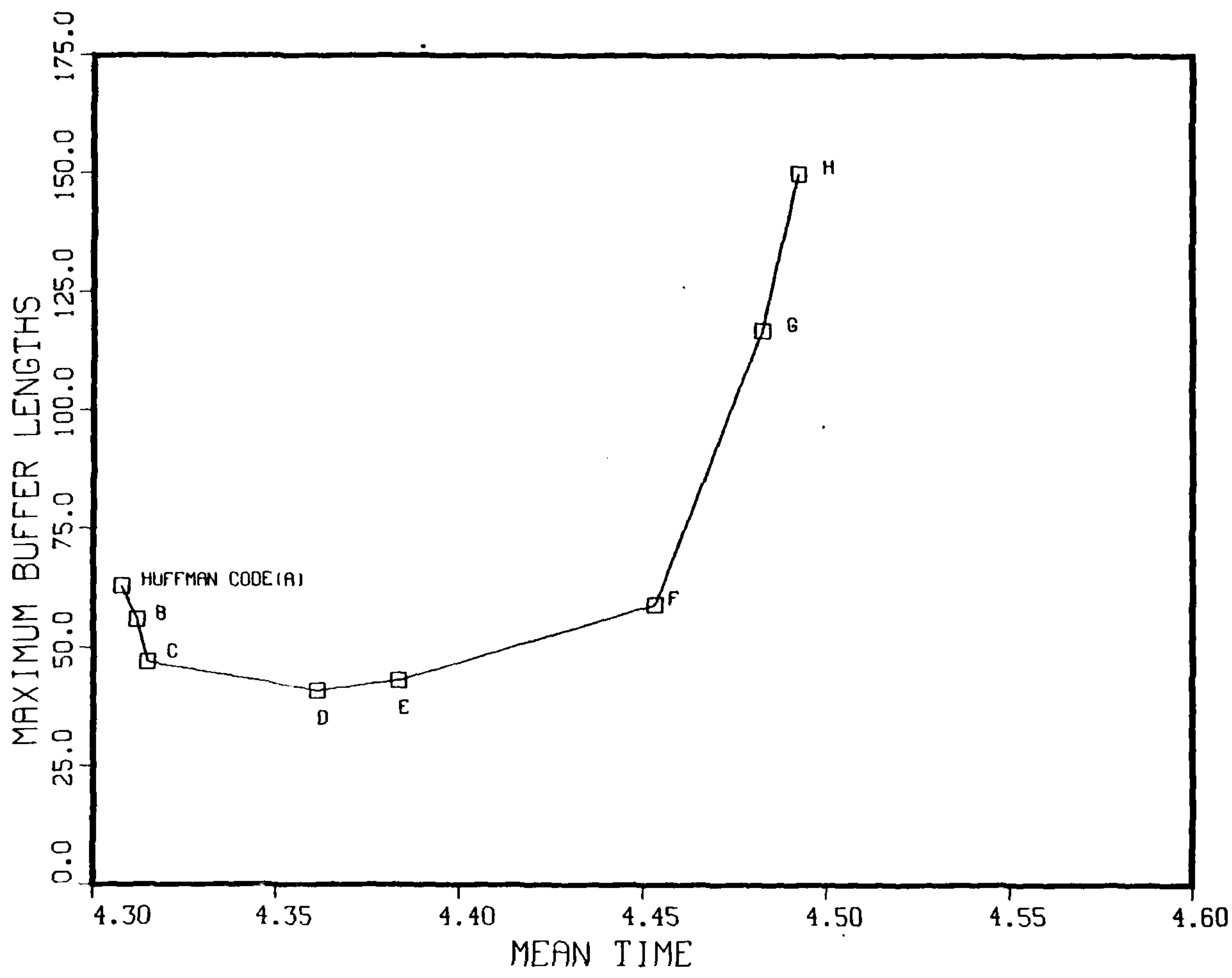


Figure 3.3 Maximum Buffer Lengths with Different Input Rates.

D. COMPARISON WITH THEORY

To be able to compare the experimental results with the theoretical results, the upper bound equation for the average wait was used [Ref. 10: page 49].
The maximum buffer length is given as :

$$\frac{[VAR(I)] + (1/m)[VAR(O)] + [(m-1)/m^2](1/C)^2}{(2/R)\ [1-(R/mC)]} \geq \text{MAX.BUF.LENGTH}$$

where;

VAR(I) = Variance of the Input Rate

VAR(O) = Variance of the Output Rate

m = Number of Servers

R = Input Rate

C = Output Rate

Since there is only one channel, by setting m=1 the above equation becomes :

$$\frac{[VAR(I)] + [VAR(O)]}{(2/R)[1-R/C]} \geq \text{MAX BUFFER LENGTH}$$

The resulting maximum buffer lengths obtained by using this equation, are given in Table 17. Figure 3.4 graphs the upper bounds of the maximum buffer lengths versus mean times which were obtained from Table 17. Once again code D requires the smallest buffer size. The shape of the curves given in Figure 3.3 and in Figure 3.4 are almost the same. These two figures emphasize how well the experimental results match the theoretical results.

TABLE 17

UPPER BOUNDS OF THE MAXIMUM BUFFER LENGTHS

INPUT RATES	MAXIMUM BUFFER LENGTHS
4.30771 (code A)	97
4.31199 (code B)	90
4.31476 (code C)	73
4.36112 (code D)	67
4.38336 (code E)	75
4.45343 (code F)	117
4.48231 (code G)	293
4.49231 (code H)	641

OUTPUT RATE : 4.5 BITS PER UNIT TIME

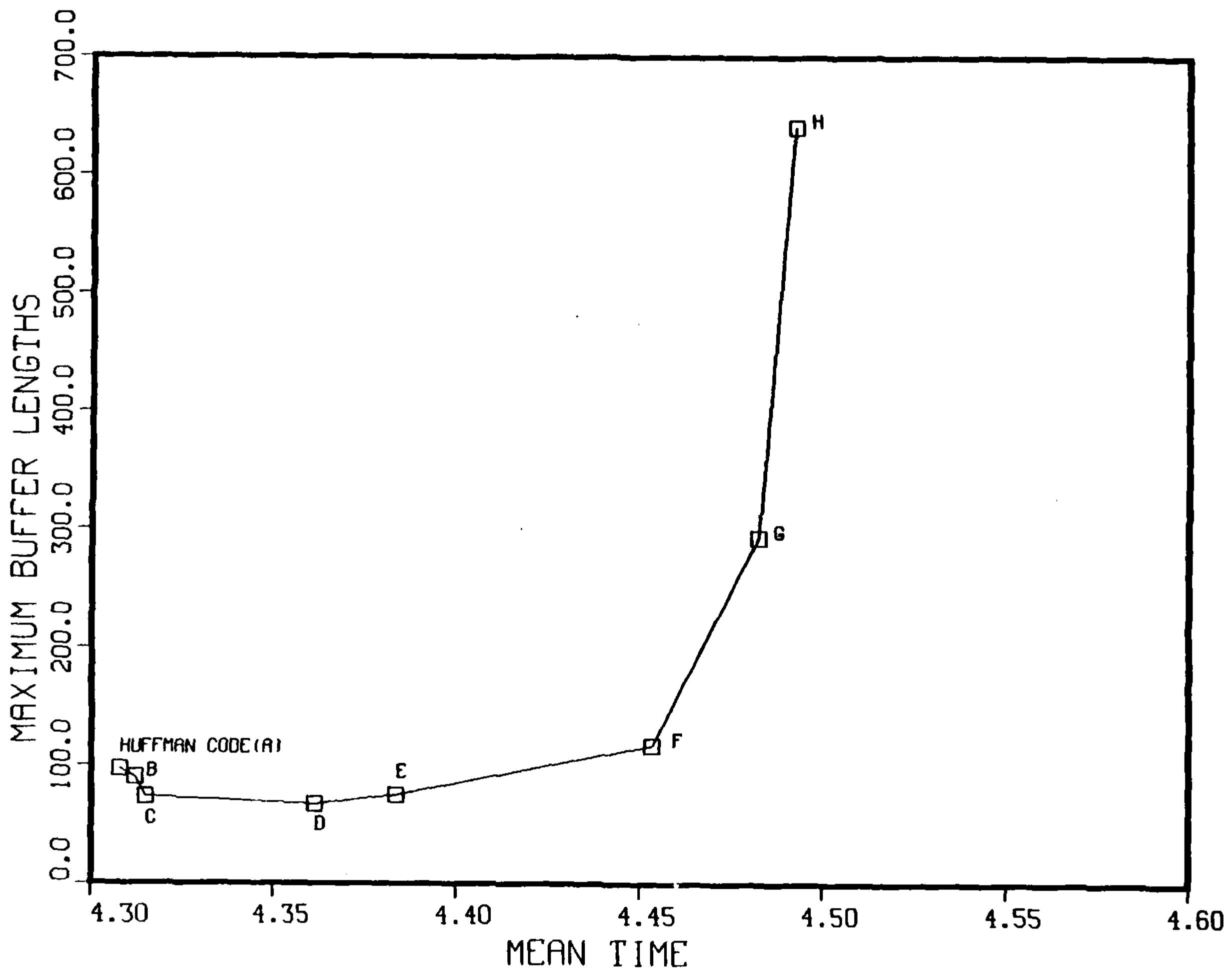


Figure 3.4 Upper Bounds of the Buffer Lengths.

E. OVERFLOW DURING TRANSMISSION

In this section we describe the number of overflows obtained using different buffer sizes. To find the number of overflows, a Fortran computer program was used. This Fortran language program which was also written by the author is given in Appendix D. This program runs with different given buffer lengths ranging from zero to a size which causes no overflow and for different lengths of messages. The results are shown in Table 18. During transmission the input rate chosen was the best code, determined in the previous section, (code D) and the output rate was kept at 4.5 bits per unit time. A graph of the buffer size versus the number of overflows is displayed in Figure 3.5 when both of the magazine articles were transmitted. It can be observed from Figure 3.5 that the provision of the larger buffer sizes results in a reduction of the number of overflows. In addition, when the given buffer length is 41, overflows do not occur. As stated in the previous section, during transmission of the two magazine articles with the same input and output rate used, the required maximum buffer length found was also 41 bits.

Table 19 shows the number of overflows for different numbers of characters when some arbitrarily chosen buffer lengths were used. Plotting this data given in Table 19, four different curves appear in Figure 3.6. This figure emphasizes that by increasing the given buffer length, the slopes of the curves approach zero. Finally after a certain value of the provided buffer size (41 bits), the number of overflows is zero for all different lengths of messages. The curve which belongs to the buffer of length 41 becomes parallel to the horizontal axis.

TABLE 18

NUMBER OF OVERFLOWS

GIVEN BUFFER LENGTHS	NUMBER OF CHARACTERS 5000	10000	12000	15000	ENTIRE ARTICLES
0	2958	6478	7873	10149	11948
1	1432	3758	4571	6188	7279
2	835	2620	3234	4543	5329
3	645	2169	2728	3887	4564
4	504	1863	2391	3446	4013
5	396	1604	2107	3076	3580
6	316	1387	1878	2771	3202
7	260	1168	1646	2477	2860
8	220	1006	1466	2223	2552
9	187	863	1313	2019	2299
10	157	751	1194	1844	2087
11	126	655	1087	1682	1891
12	112	589	1018	1569	1765
13	97	492	914	1422	1604
14	80	421	836	1309	1473
15	64	334	741	1170	1314
16	55	272	664	1054	1182
17	40	197	566	890	1006
18	23	147	501	767	861
19	7	111	447	666	737
20	3	84	394	563	620
21	1	57	351	486	536
22	0	40	310	426	475
23	0	34	273	366	412
24	0	25	230	301	344
25	0	20	200	249	290
26	0	15	172	207	244
27	0	14	145	171	205
28	0	13	125	141	171
29	0	13	103	117	145
30	0	12	95	105	122
31	0	9	89	96	103
32	0	5	75	81	84
33	0	3	61	63	64
34	0	0	45	46	47
35	0	0	37	37	37
36	0	0	29	29	29
37	0	0	24	24	24
38	0	0	13	13	13
39	0	0	4	4	4
40	0	0	1	1	1
41	0	0	0	0	0

INPUT RATE : 4.36112 OUTPUT RATE : 4.5

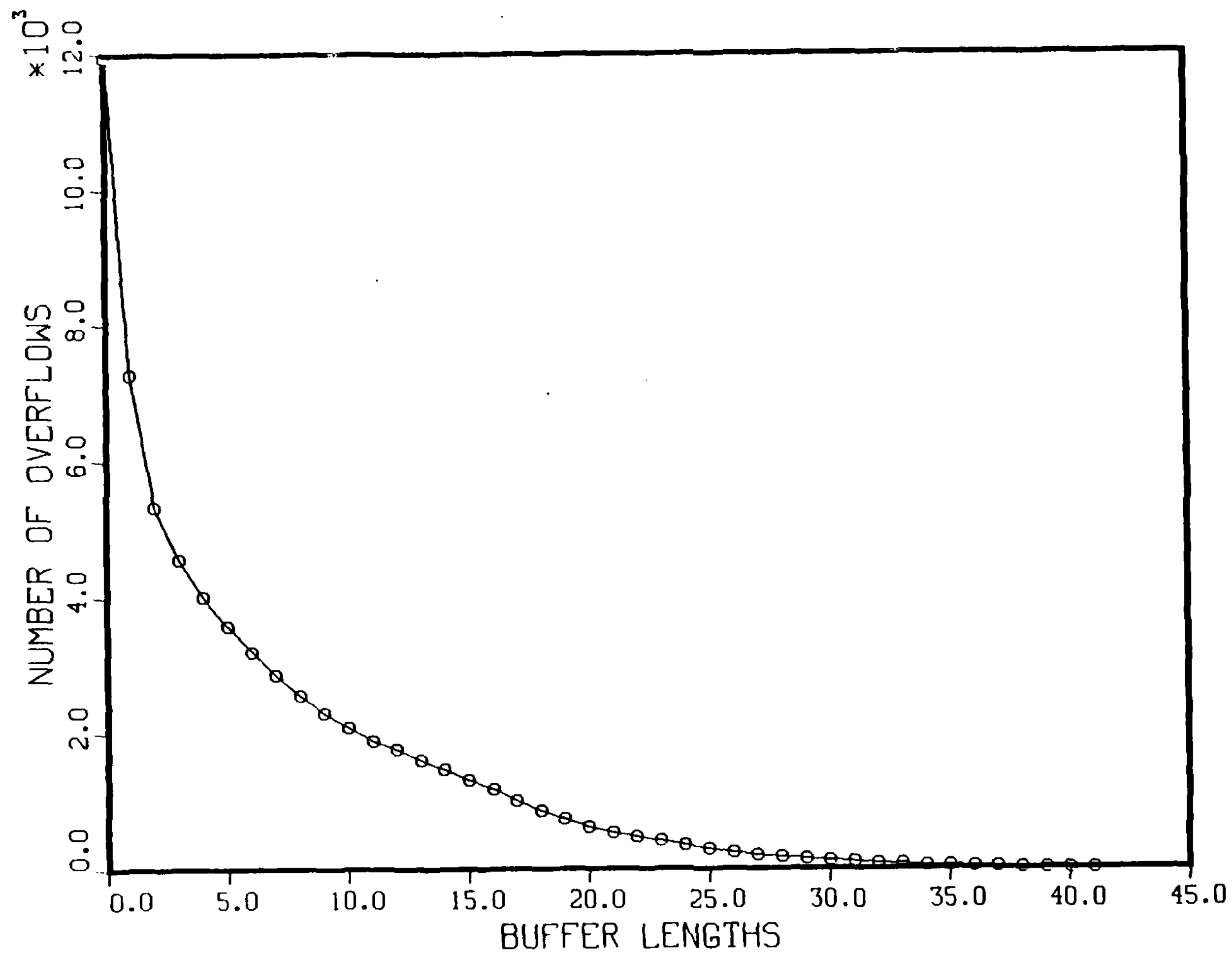


Figure 3.5 Number of Overflows with Given Buffer Lengths.

TABLE 19

OVERFLOWS WITH DIFFERENT MESSAGE LENGTHS

	PROVIDED BUFFER LENGTHS			
MESSAGE LENGTHS	0	10	20	41
1000	662	145	3	0
3000	1796	149	3	0
5000	2958	157	3	0
8000	5047	338	14	0
10000	6478	751	84	0
12000	7873	1194	394	0
15000	10149	1844	563	0
ENTIRE ARTICLES	11948	2087	620	0

INPUT RATE : 4.36112 BITS PER UNIT TIME

OUTPUT RATE : 4.5 BITS PER UNIT TIME

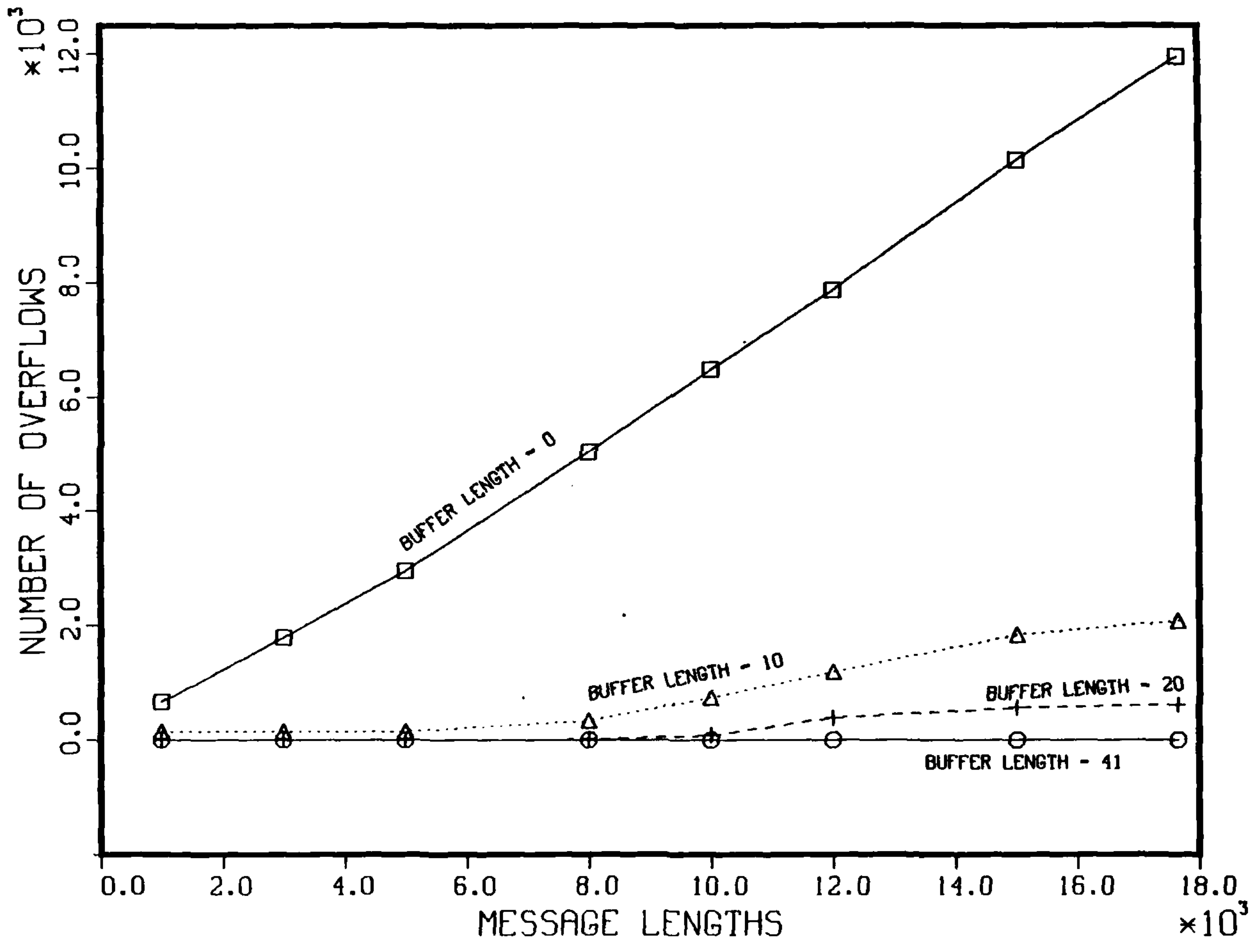


Figure 3.6 Number of Overflows with Different Message Lengths.

IV. A POSSIBLE DESIGN FOR A PRACTICAL SYSTEM

A. PROBLEM

We considered forwarding information either from here to there (transmission) or from now to then (storage). We started with a source of information (symbols) and encoded it in a fashion such as discussed in the previous chapters. The information was then sent through a channel. Next the message will be decoded and finally the recovered message will be transmitted to its destination. The problems then are: How can the variable length codes be decoded at the receiver? What is the performance difference between decoding a block code and variable length code?

B. SOLUTION

The first property that is needed for decoding is unique decodability. This means that the received message has to have an unique description. The second property that is needed is instantaneous decodability. As an example of instantaneous decodeable codes consider an alphabet with five letters and their corresponding code words. See (Table 20).

The receiver establishes a decision tree in order to decode a message using this code [Ref. 1: page 53]. Starting with the first decision point (initial state), the first binary digit arriving at the decoder causes a branch, either to a terminal state S1 if the received digit is 0, or to a second decision point if it is a 1. If the second binary digit received is 0, the second branch goes to the terminal state S2. If the second digit is a 1, then the second branch goes to the third decision point. This continues until the fourth digit reaches the receiver. In this case the fourth decision point goes to the terminal state S4 if that digit is a 0, and to the terminal state S5

TABLE 20
EXAMPLE FOR INSTANTANEOUSLY DECODABLE CODES

SYMBOL	CODE WORDS
S1	0
S2	10
S3	110
S4	1110
S5	1111

if it is a 1. When a terminal node is reached, the process begins again at the initial decision point. Each bit of the received message is examined only once. Therefore, the decoding in this example is instantaneous since, when a complete encoded symbol is received, the decoder knows which symbol was transmitted. In an instantaneous code, no code word can be a prefix of another code word. Alternatively, for those codes, which have some code words as the beginning part of some other code words, the receiver is not able to identify immediately which code word is received. In this case the codes can be uniquely decoded, but they are not instantaneous. One of the possible way to decode the symbols in a uniquely decodeable but not instantaneous code is to begin to decode from the back end of the received message.

A necessary and sufficient condition for the existence of an instantaneous code is given by the Kraft inequality [Ref. 1: page 57]. It was observed that the coding systems given in Table 10 agreed with this inequality, when they were tested. These variable length codes can be decoded by using a finite automaton (decision tree) algorithm, since they are instantaneously decodable codes.

C. EFFECTIVENESS

The major significance of channel capacity is its relationship with the bandwidth. We showed that there will be no need for a buffer when transmission is accomplished consisting of 6 bits per unit time both for the input and output rate. According to the theoretical results, if the channel bandwidth was decreased to 4.5 bits per unit time, (giving a saving of 25% in the bandwidth) there will be a necessity to have a finite length buffer during transmission of an infinite length message. This means that the transmitted message can be decoded at the receiver completely after a delay time equal to the time required to empty the buffer. Evidently, there will be a time delay to recover variable length coded messages, since the excess digits should be put in a buffer.

As far as the base performance is concerned, it can be assumed that only 1 unit of the channel is used. On the other hand, when experimental codes from A to M, (given in Table 12) were sent, only 0.75 unit of this channel was used, but now an extra buffer was required. Code D, which requires the minimum buffer among all other coding systems, was selected for that reason to compare with the base performance. Using the code D the required buffer length is 67 for the transmission of an infinite length message, but is only 41 when the articles given in Appendix A were transmitted. In the worst case it takes only 67 unit time delay to decode an infinite length message with code D. Bear in mind that in real applications, a message can not be infinite. Therefore, the time delay to decode the variable length coded messages is shorter than the delay to decode an infinite one.

Although it seems that the decoding of a variable length code is not as effective as the block code decoding when the time delay is considered, there are some advantages to using a variable length code over block coding. First, to evade

the discrimination of the transmitted message by an unintended recipient, the variable length code becomes more difficult to decrypt than the block code. Second, there is only so much bandwidth in the spectrum of available frequencies that passes through the earth's atmosphere, and already much of it is assigned to various uses. Therefore, the proposition of saving even 25% from the bandwidth can be observed as a valid estimated performance criteria.

Accordingly, these two important properties of the variable length codes carry an important role for military applications. However, the defect for that use is the time delay during the decoding of the received messages. In a critical case when transmitting an urgent short message the negative aspect of the time delay loses its importance, since the lengths of the buffer grow with the length of the messages. On the other hand, when longer messages are transmitted, considering both jamming avoidance and the time delay at the decoder, the bandwidth can be increased to higher rates to obtain smaller buffer lengths.

Definition of some other parameters, which would obtain better modified Huffman codes could result with a reduction of the lower bounds given in Figure 2.1, 2.2 and 2.3. Thus, with these coding systems, if done properly, more effective practical systems could be designed.

V. CONCLUSIONS

The minimization of the average code length by using Huffman coding, produces a large variance, resulting in a large variable code, which in turn causes a need for large buffer size. It was shown that by losing a little in the mean time, much can be acquired in the reduction of the variance. Thus, the size of the buffer can also be decreased with these lower variance codes. The manipulation of the modified Huffman codes (experimental codes) causes a gain in the bandwidth, when compared with the block encoding. But, this also results in larger buffers, which produces a time delay to recovery of the received messages. The size of the buffer can be decreased by increasing the transmission rate (bandwidth) of the experimental codes up to the output rate of the base performance.

For the two reasons given in the beginning of Chapter II, the optimization of only average code length was not considered during the progress of this research. The probability distribution shown in the same chapter reflects the frequencies of the texts given in Appendix A. These frequencies can always be different by using various texts. Therefore, only the experimental results were included during the work.

This research also indicates that optimization of a subsytem is sometimes less important than the optimization of the entire system. As a rule, total system performance can be degraded when only a particular aspect is concerned.

APPENDIX A

THE TURKISH MAGAZINE ARTICLES AND PROGRAMS

1. THE MAGAZINE ARTICLES

The first article titled "Strange Shapes of Modern Ships" is given below.

BIR DERGININ RESSAMI, EN GUCLU VINCLERIN YAPAMADIGI ISI BASARARAK, 50.000 TONLUK BIR "OKYANUS DEVI"NI SUDAN CIKARDI VE BOYLECE, GEMININ BURNUNDAKI YUMRUBAS "BALB" ORTAYA CIKMIS OLDU. GEMININ KIC TARAFINDA DA BAZI YENILIKLER GOZE CARPIYORDU. BUNLARIN SIRRI ACABA NE OLABILIRDI? OTOMOBIL YAPIMCILARININ YENI GELISTIRDIKLERI MODELLERI DENEDIKLERI "RUZGAR TUNELLERI"NIN BIR BENZERI DENIZ TEKNELERI UZERINDE CALISAN MESLEKTASLARI ICIN DE GECERLI OLUYOR. ONLARIN DA YENI TEKNE MODELLERI DENEDIKLERI "TEST HAVUZLARI" VAR. YENI GEMILER, ANCAK, BU HAVUZLARDA YAPILAN DENEYLERIN OLUMLU SONUCLAR VERMESINDEN SONRA, INSA EDILMEK UZERE KIZAGA KONUYOR. BU ARADA, GEMI MUHENDISLERININ ISLERI, KARA ARACLARI UZERINDE UGRAS VEREN MESLEKTASLARININ ISLERINDEN BIRAZ DAHA GUC. BU GUCLUK, DAHA MODEL ASAMASINDA BASLAR. DENEYLERI YAPILAN GEMI MODELLERI, YETERINCE BUYUK OLDUGU ZAMAN, DENEYLERDEN ALINAN OLCUM SONUCLARI, ISTENILENI VEREBILMEKTEDIR. GUCLUGU YARATAN IKINCI ETKEN DE, DUNYAMIZIN "SU" VE "HAVA" OLARAK BILINEN IKI ELAMANINDAN KAYNAKLANMAKTADIR. BIR KARA TASITINDA, KAROSERI SADECE RUZGARA KARSI KOYMAK ZORUNDA OLMASINA KARSIN, BIR TEKNENIN HEM DALGAYA VE HEM DE, RUZGARA KARSI KOYMASI GEREKIR. ESKI TARIHLERDE INSA EDILMIS GEMILERDE, BURUNLAR KESKINLESTIRILIR VE BOYLECE SUYUN DAHA AZ BIR DIRENIMLE YARILMASI SAGLANIRDI. ANCAK, BU IS, ASLINDA HIC DE GORUNDUGU KADAR BASIT DEGILDIR. GEMI HESAPLARI, SUALTINDAN ATESLENEN BIR ROKETIN HESAPLARINDAN DAHA KARMASIK VE GUCTUR. BIRAZ ONCE

BELIRTTIGIMIZ GIBI BIR GEMI, SU VE HAVA ORTAMINDA SEYREDER. BU NEDENLE DE, OZELLIKLE HAVANIN VE SUYUN BIRLESTIGI NOKTA, MUHENDISLER ICIN BIR "BILMECE"DIR. DENEY HAVUZLARINDAN ALINAN SONUCLAR OKYANUSLAR ICIN DE GECERLI OLDUGUNDAN; BU BENZER ILISKILERDEN YARARLANAN GEMI MUHENDISLERI, DENEYLERINI DENEY HAVUZLARINDA YAPMAKTADIRLAR. GEMIYE HAREKET VEREN PERVANE, TEKNEYI ILERIYE ITERKEN, GEMININ BURNUNDA BIR DALGA OLUSUR. BU DALGA, BURUNDA, YANLARDA, DIPTE VE KICTA GEMIYI YALAYARAK GECER. ANCAK, ANILAN DALGA ALISILAGELEN TIPTE BIR DALGA OLMAYIP, SAGA-SOLA KARISIK HAREKETLER YAPAN SULAR HALINDEDIR. GEMI BURNUNDA OLUSAN VE TEKNE TARAFINDAN ILETILEN BU SU KITLELERI, GEMI BURNUNUN GENISLIGI ORANINDA ARTAN BIR YIGILMA YAPARAK, ISTENILMEYEN BIR DIRENC OLUSTURUR (SEKIL 1). ISTENILMEYEN BU DIRENCIN ETKISINI AZALTABILMEK ICIN, GEMININ BURNUNDA YUMRUBAS DENILEN VE MAHMUZU ANDIRAN BIR CIKINTI YAPILIR. YUMRUBASIN ETKISI SOYLE ACIKLANABILIR: YUMRUBASLI BIR TEKNE, ONUNDE IKI DALGA TEPESI OLUSTURUR. BUNLARDAN, TEKNENIN OLUSTURDUGU DALGA TEPESI, YUMRUBASIN OLUSTURDUGU DALGANIN CUKURUNU DOLDURARAK, GEMI BURNUNDAKI YIGILMAYI ONLER (SEKIL 2). SONUC OLARAK DA, ISTENILMEYEN DALGA YOK EDILIR. YUMRUBAS ADI VERILEN BU YENI BURUN TIPI, AMERIKALI GEMI ADAMI DAVID TAYLOR'UN BULUSUDUR. YUZYILIMIZIN BASLARINDA TAYLOR, YUMRUBASLI GEMILERIN, DIGERLERINE KIYASLA DAHA KUCUK DALGALAR OLUSTURDUGUNU TESPIT ETMIS VE BUNUN TEORISI DAHA SONRA GELISTIRILMISTIR. ANCAK, TUM OLASILIKLARI AYDINLIGA KAVUSTURACAK KESIN FORMULLER GUNUMUZDE DAHI TAM OLARAK SAPTANMIS DEGILDIR. YUMRUBAS TEORISININ GELISMESINI ASAGIDAKI MADDELERLE ACIKLIYABILIRIZ: 1. SEYIR HALINDEKI BIR GEMI, ONUNDE BUYUK BIR DALGA TEPESI OLUSTURARAK ILERLER. 2. SU YUZEYININ HEMEN ALTINDA HAREKET ETTIRILEN BIR KURE, ARKASINDA BIR DALGA CUKURU OLUSTURUR. 3. GEMI MODELININ BURNUNA BIR KURE YERLESTIRILEREK, KURENIN OLUSTURDUGU DALGA CUKURU ILE GEMI MODELININ OLUSTURDUGU DALGAYI CAKISTIRACAK BIR DENEY UYGULAMASI GERCEKLESTIRILIR. 4. DENEYDE, DALGA

CUKURUNUN DALGA TEPESINI YUTTUGU GORULUR. 5. DALGA TEPESI YUTULDUGUNDAN; ISTENILMEYEN DIRENC ETKISINI KAYBEDER. SONUC OLARAK, GEMI MODELI DAHA BUYUK BIR HIZ KAZANIR VEYA HAREKETI ICIN GEREKLI OLAN GUC AZALIR. ALINAN BU SONUC, GEMININ TUKETTIGI YAKITTA HIC DE AZIMSANMAYACAK BIR TASARRUF SAGLANDIGINI ORTAYA KOYAR. ARMATORLERIN YUMRUBASLI GEMI SIPARISLERINE AGIRLIK VERMELERINDEN SONRA, MUHENDISLERIN ISLERI DAHA DA GUCLESMISTIR. ILK ZAMANLARDA YUMRUBASLAR, YOLCU VE SAVAS GEMILERINDE UYGULANIYORDU. BUNUNDA NEDENI, ANILAN GEMILERIN SEFERLERINI GENELLIKLE SABIT BIR SU KESIMINDE YAPMALARI IDI. OYSA, ARMATORUN SIPARISE BAGLADIGI YUK GEMILERINDE SU KESIMI (DRAFT), GEMILERIN YUKLU VEYA BOS OLMALARINA GORE, DEGISEBILDIGI ICIN, GEMI BURNUNDA YER ALAN YUMRUBAS, ETKINLIK POZISYONUNU KORUYAMAMAKTADIR. GEMI, YUKUNU ALARAK SEFERE CIKTIGINDA; YUMRUBAS, SUALTINDA, KALARAK, ETKINLIGINI SURDURMEKTE ISE DE, YUKUN BOSALTILMASINDAN SONRA, SU YUZEYINE CIKMAKTA VE SONUC OLARAK, ETKINLIGINI KAYBETMEKTEDIR. BU DURUM, YUMRUBASIN GEMI BURNUNDA NEREDE YER ALMASI GEREKTIGI SORUNUNU ORTAYA CIKARMISTIR. DAHA SONRA, YUMRUBAS, GEMI BURNUNUN BIRAZ DAHA ASAGISINA ALINARAK, SUYUN ALTINDA BIRAKILMIS VE ISTENILEN SONUCA KISMEN DE OLSA ULASILMISTIR. YUMRUBASI SADECE SUALTINDA BIRAKMAKLA SORUNLARA COZUM GETIRILEMEMEKTEDIR. CUNKU, HER TEKNE KENDINE OZGU BIR DALGA SEKLI OLUSTURMAKTA VE BU NEDENLE DE, YUMRUBASIN, KULLANACAGI TEKNE ILE UYUM SAGLAYACAK OZELLIKLERE SAHIP OLMASI GEREKMEKTEDIR. GEMI MUHENDISLERININ GOGUSLEMEK ZORUNDA OLDUKLARI BU GUCLUKLER, YENI ARASTIRMA ALANLARININ DOGMASINA YOL ACMIS VE BU KEZ DE, ARASTIRMALAR GEMININ KIC TARAFINDA YOGUNLASMISTIR. YAKLASIK 20 YIL KADAR ONCE, HAMBURGLU GEMI MUHENDISI ERNST NONNECKE, YENI BIR KIC FORMU GELISTIRMIS ISE DE, ONUN BU BULUSU ANCAK SON YILLARDA DEGER KAZANMAGA VE DIKKAT CEKMEGE BASLAMISTIR. NITEKIM, NONNECKE'NIN BULUSU, BIR KORE TERSANESINDE 2 KONTEYNER GEMISINDE UYGULAMAYA KONULMUSTUR. TEORIK CALISMALAR HAMBURG'DA BASLAMIS VE BUNU IZLEYEN DENEYLERDE,

INSA EDILECEK GEMININ BIR MODELI, BOYU 300 M. VE DERINLIGI 18 M. OLAN BIR DENEY HAVUZUNA CEKILEREK, NONNECKE'NIN GELISTIRDIGI KIC FORMUNUN USTUNLUGU KABUL EDILMISTIR. BU TIP ASIMETRIK KIC FORMU: SANCAK TARAFI CUKUR VE ISKELE TARAFI DISA DOGRU BOMBELIDIR. BU FORMUN OZELLIGI, SUYUN AKISINI DUZELTEREK, DOGRUDAN PERVANEYE VERMESIDIR. NONNECKE TIPI KIC TEORISI SU SEKILDE ACIKLANABILIR: SIVI ICINDE HAREKET EDEN BIR GOVDE, SUYU BAS TARAFTAN YARAR. YARILAN SU, GOVDENIN KIC TARAFINDA YINE BIRLESMEK EGILIMI GOSTERIRKEN, BU KEZ DE GEMININ PERVANESI ILE KARSILAR. GEMININ HAREKET YONUNE GORE, SAGA DOGRU DONEN PERVANE, SUYU TEKNENIN SANCAK (SAG) TARAFINDAN ASAGIYA ITER, BUNA KARSIN, ISKELE TARAFINDAN (SOL), YUKARIYA DOGRU ITILEREK, TEKNENIN KIC TARAFINDA BIRLESME EGILIMI GOSTEREN SU, BIRLESMEDEN PERVANENIN AKIMINA KAPILIR. CEKILEN SUALTI FOTOGRAFLARI ILE TESPIT EDILEN BU OLAY, SUYUN GEMIDE ISKELE TARAFINDAN GEREKTIRDIGI ITICI GUCU OLUSTURMADAN, YUKARIYA DOGRU ITILDIGI GERCEGINI ORTAYA KOYMUSTUR. BU OLAY UZERINDE DURAN NONNECKE, ISKELE TARAFINDAN PERVANEYE YONELEN SU AKISINI DUZENLEYEBILMEK ICIN GEMIDE SANCAK VE ISKELE TARAFLARININ PERVANEYE YAKIN OLAN KISIMLARINDA, TASARLADIGI FORM DEGISIKLIKLERINI GERCEKLESTIRMISTIR. BUNA GORE, GEMININ SANCAK TARAFI CUKURLASTIRILMIS; ISKELE TARAFINDA ISE, CUKURLUGUN YERINI YUMUSAK BIR BOMBE ALMISTIR (SEKIL 5). SONUC OLARAK, SUYUN DAGILMAKSIZIN VE TURBULANSA UGRAMAKSIZIN, PERVANEYE AKABILMESI SAGLANMISTIR. SEKIL 3 VE 5 ESKI VE YENI TIP IKI GEMININ EN KESIT EGRILERINI VERMEKTEDIR. ESKI TIP BIR GEMIDE EN KESIT EGRILERI SIMETRIK BIR BICIM GOSTERMEKTE VE GEMININ ORTASINDA DUZ BIR CIZGI BOYUNCA BIRLESMEKTEDIR (SEKIL 3). DIGER TIP KIC FORMUNDA ISE, ANILAN EGRILER ASIMETRIK OLARAK GELMEKTE VE GEMININ ORTASINDA "S" SEKLINDEKI BIR CIZGI UZERINDE TOPLANMAKTADIR (SEKIL 5). SEKIL 4 VE 6'DA, ESKI VE YENI TIP KIC FORMLARININ BIRER PROFILI ILE PERVANEYE DOGRU YONELEN SUYUN AKISI GORULMEKTEDIR. ESKI TIP KIC FORMUNDA (SEKIL 4); PERVANEYE

DOGRU AKIS YAPAN SU, PERVANE ILE KARSILASTIGINDA TURBULANSA UGRAMAKTA VE DOLAYLI OLARAK DA, GEMI DIESELININ PERVANEYE AKTARDIGI GUCTE KAYBA YOL ACMAKTADIR. NONNECKE TIPI KIC FORMUNDA ISE, PERVANEYE YONELEN SUYUN AKISI DUZENLENMIS (SEKIL 6) VE DUZENLENEN SU, TURBULANSA UGRAMADAN, PERVANE TARAFINDAN ITILEREK, PERVANENIN VERIMI ARTIRILMIS VE GEMININ DAHA AZ BIR GUCLE DAHA BUYUK BIR HIZ KAZANMASI SAGLANMISTIR. "THEA S" ADLI 124 METRELIK GEMIDE YAPILAN DENEYLER, BU YENI KIC FORMUNUN GUNDE 2.000 LITRELIK BIR YAKIT TASARRUFU SAGLADIGINI ORTAYA KOYMUSTUR. ESKI TIP GEMI FORMLARININ GECERLI OLDUGU GUNLERE KIYASLA, YAKIT FIATLARININ BUGUN 10 KAT ARTTIGI GOZ ONUNDE TUTULURSA, GEMILERE SAGLANAN YAKIT TASARRUFUNUN NE KADAR ONEMLI OLDUGU VE MODERN GEMILERININ NICIN BOYLE GARIP BICIMLERDE INSA EDILDIGI SORUSU KENDILIGINDEN AYDINLIGA KAVUSABILIR.

The second magazine article titled "Story of the Space Shuttle" is given below.

1970'LERE DEK DAYANAN UZAY MEKIGI PROJESININ TEMEL AMACI, UZAYA DAHA UCUZ VE DOLAYISIYLA DAHA SIK GITMEKTIR. MEKIKTEN ONCE UZAYA ATILAN INSANLI VE INSANSIZ UYDULAR, SONDA VE ROKETLER SADECE BIR KEZ KULLANILABILIYORDU VE BU NEDENLE MALIYETLERI YUKSEK OLUYORDU. UZAY MEKIGI PROJESI ILE INSANOGLU, AYNI UZAY ARACINI SUREKLI KULLANMA OLANIGINA KAVUSTU. BU PROJENIN EN BELIRGIN OZELLIGI UCAK TEKNOLOJISI ILE UZAY TEKNOLOJISINI BIR ARAYA GETIRMESIDIR. SISTEM GENELDE UC ANA BOLUMDEN OLUSMAKTADIR: 1) YORUNGE ARACI DA DENEN UZAY GEMISININ KENDISI; 2) BUYUK DIS YAKIT TANKI; 3) DIS YAKIT TANKININ HER IKI TARAFINDA BULUNAN KATI YAKITLI ROKETLER. SISTEMI FIRLATMA ANINDA, GEMININ ARKASINDA BULUNAN ANA MOTORLAR VE IKI FIRLATICI ROKET ATESLENIR. BU ISLEMIN SONUNDA, OTUZ MILYON NEWTON'LUK COK BUYUK BIR FIRLATMA KUVVETI, SISTEMI HAVALANDIRIR. HAVALANDIKTAN BIR DAKIKA SONRA SISTEMIN SURATI, SES SURATINI ASAR. BU SIRADA GEMININ ICINDE OLSANIZ VE KENDINIZI TARTSANIZ, YERYUZUNDE 60 KILO

GELEN VUCUDUNUZUN, IKI DAKIKA ICINDE SISMANLAMIS OLMAMASINA KARSIN, 180 KILO GELDIGINI GORURSUNUZ. BU ILGINC DURUM, ARACIN IVMESININ, CEKIM IVMESINDEN UC KAT FAZLA OLMASINDAN KAYNAKLANMAKTADIR. HAVALANDIKTAN SONRA KATI YAKITLI ROKETLERIN YAKITLARI BITER VE DIS YAKIT TANKINDAN AYRILIRLAR. BU ANDA GEMI, 50 KM. YUKSEKLIKTE VE HIZI SAATTE 5.000 KM'YE ULASMISTIR. AYRILAN ROKETLER, ILK HIZLARINDAN DOLAYI DERHAL ASAGIYA DUSMEZLER. 50 KM'DE AYRILAN BU ROKETLER, 67 KM'YE DEK CIKAR VE SONRA DUSMEYE BASLAR. DUSERKEN, YUZEYDEN YAKLASIK 3 KM. YUKSEKLIKTEN, UC EVRELI PARASUT SISTEMI CALISIR VE DUSUSUN HIZINI AZALTIR. DENIZE DUSEN ROKETLER, SU YUZEYINE DEGDIKLERI ANDA PARASUTLERDEN AYRILIR VE ALT TARAFTA BULUNAN OZEL BOLMELER SISEREK, ROKETLERIN BATMAMALARI SAGLANIR. DAHA SONRA BUNLAR DENIZDEN TOPLANIR. GEREKLI ONARIM VE BAKIM YAPILARAK, BIR SONRAKI UCUS ICIN HAZIRLANIRLAR. BU KATI YAKITLI ROKETLERIN KALKISTAKI AGIRLIGI, YAKLASIK 580 TONDUR VE 11.800.000 NEWTON'LUK BIR ITME MEYDANA GETIRMEKTEDIR. UZUNLUGU 45.5 METRE, SILINDIRIK GOVDENIN CAPI ISE 3.7 METREDIR. UZAY GEMISININ ANA MOTORLARINA YAKIT VEREN BUYUK DIS TANK ISE YERDEN 200 KM. YUKSEKLIKTE IKEN YAKITI BITTIGINDE ARACTAN AYRILIR. 20 KATLI BIR APARTMAN YUKSEKLIGINDE (50 M.) OLAN BU BUYUK SILINDIRIK TANKIN CAPI 30 METREDIR. YAPIMI ICIN 30 TON ALUMINYUM KULLANILAN BU TANKIN BIR KEZ KULLANILMASI, BIR COK KISININ NASA'YI ELESTIRMESINE NEDEN OLMAKTADIR. CUNKU MEKIKTEN AYRILAN TANK, DAHA SONRA DUNYA ATMOSFERINE GIREREK YANMAKTADIR. NASA MUHENDISLERI BU TANKLARDAN NASIL YARARLANACAKLARINI DUSUNMEKTEDIRLER. HAZIRLANAN BU PROJEYE GORE, 1990'DAN SONRA KURULMASI BEKLENEN UZAY ISTASYONUNUN, BU TANKLARDAN YIRMISININ BIR ARAYA GETIRILEREK YAPILMASI ONERILMAKTEDIR. MARTIN MARIETTA AEOROSPACE SIRKETI'NIN GELISTIRILMIS PROGRAMLAR BASKANI OLAN FRANK WILLIAMS'A GORE GEMI, TANKINI UZAYDA BIRAZ DAHA SONRA BIRAKACAK. O ZAMAN TANK, YER ATMOSFERINE DUSMEYECEK, GEMIYI IZLEYEREK ISTENEN YORUNGEYE OTURTULMASI SAGLANACAK. DENEYLERIN YAPILACAGI VE

ICINDE RAHATCA YASANILABILECEK SAGLAMLIKTA OLAN BU SILINDIRLER UC UCA EKLENDIGINDE, ISTENEN UZAY ISTASYONUNUN HEM DAHA KISA ZAMANDA, HEM DE DAHA EKONOMIK BIR SEKILDE YAPILABILECEGI ILERI SURULUR. UZAY GEMISININ ON GOVDESI VE MURETTEBAT BOLUMU, ALUMINYUMDAN YAPILMIS UC KATTAN OLUSMAKTADIR. EN UST KATTA, YORUNGE ARACININ KENDISINI, TUM UZAY GEMISI SISTEMINI VE TASINAN YUKU YONETEN, DENETLEYEN KUMANDA SISTEMI YER ALMAKTADIR. BU KATTA, UC ASTRONOT ISKEMLESI BULUNMAKTADIR. ORTA KAT, UCUS ZAMANI TASIMA VE YASAM BOLUMU OLARAK AYRILMISTIR. AYRICA BU BOLUM, GEMININ YUK TASIYAN KARGO BOLUMU ILE BAGLANTILIDIR. ALT KATTA ISE CEVRE KONTROL GERECLERI YER ALMAKTADIR. GEMININ ORTA BOLUMU, YUK TASIYAN KARGO BOLUMUDUR VE UZAYA GIDERKEN USTTEN ACILAN IKI KAPAK ILE ORTULMEKTEDIR. UZAYDA BU KAPAKLAR ACILARAK, UYDULARI YORUNGEYE OTURTMAK, YURUYUS YAPMAK GIBI CESITLI GOREVLER YERINE GETIRILMEKTEDIR. ARKA GOVDE VE MOTOR YUVALARINI TASIYAN SON BOLUM, YORUNGE ARACININ EN KARMASIK PARCASIDIR. SADECE 8 DAKIKA SUREYLE ATESLENEN VE YORUNGEYE ERISMEZDEN ONCE 6 MILYON NEWTON'LUK FIRLATMA KUVVETI YARATAN UC ANA MOTOR BU BOLUMDEDIR. ANA MOTORLAR SUSTUKTAN SONRA GEMIYI YORUNGESINE OTURTAN IKI ROKETTEN OLUSAN YORUNGE MANEVRA SISTEMI DE BU ARKA BOLUMDEDIR. SON OLARAK BU BOLUMDE 38'I ANA, 6'SI DUYARLI OLMAK UZERE TOPLAM 44 KUCUK ROKETTEN OLUSMUS, TEPKI-DENETIM SISTEMI BULUNMAKTADIR. BU SISTEM, ARACIN (YORUNGE ICINDE KALMA KOSULU ILE) KONUMU VE UC EKSENI BOYUNCA DONME HAREKETLERI SAGLAMAKTADIR. YUKARIDA KISACA OZELLIKLERINI TANITMAYA CALISTIGIMIZ UZAY GEMISI ILK UZAY UCUSUNU, 3 YILLIK GECIKMEDEN SONRA, 1981 YILINDA YAPTI. UCUSA HAZIRLANAN 4 UZAY GEMISINDEN ILK YAPILANI, COLOMBIA ADINI TASIYORDU. UCUS KOMUTANI VE PILOT, ILK GEMI SEYRININ PERSONELIYDILER. 12 NISAN 1981 GUNU COLOMBIA FLORIDA'DAKI FIRLATMA USSUNDEN HAVALANDI. DUNYA CEVRESINDE 36 TUR ATAN GEMI KALKISTAN 54.5 SAAT SONRA, 14 NISAN GUNU YERYUZUNE DONDU. UCUS BASARILI GECMISTI AMA; GEMIYI YUKSEK SICAKTAN KORUYAN KORUMA FAYANSLARI ONEMLI DERECEDE HASARA UGRAMISTI.

HASAR NEDENI OLAN SICAKLIK, OZELLIKLE ARAC DUNYA'YA DONERKEN, ATMOSFERDEKI SURTUNMEDEN KAYNAKLANIYORDU. IKINCI UCUS, 14 KASIM 1981 GUNU GERCEKLESTIRILDI. BES GUN OLARAK DUSUNULEN UCUS PROGRAMI YARIDA KESILDI VE GEMI IKI GUN SONRA YERYUZU'NE DONDU. BU UCUSUNDA HAVA KIRLILIGI, DENIZ ARASTIRMALARI GIBI BIR TAKIM BILIMSEL ARASTIRMALAR YAPILDI. AYRICA, KANADALILARIN YAPTIGI HERHANGI BIR YONE DOGRU 15.6 METRE UZANABILEN, GEMI DISINDAKI BIR NESNEYI TUTMAK ICIN VEYA ICINDEKI BIR ALETI TUTUP UZAYA BIRAKABILMEK ICIN KULLANABILECEK, KIMININ VINC, KIMININ ROBOT, BAZILARININ DA MEKANIK KOL DEDIGI BIRIMI DENEDILER. BU UCUSTA GEMI, BIRINCIYE GORE DAHA AZ HASARA UGRAMISTI. UCUNCU UCUS, 22 MART 1982 GUNU BASLADI VE ILK KEZ SEKIZ GUN SURDU. GEMI, PLANLANAN SEYRINI BIR GUN GECIKMEYLE 30 MART'TA TAMAMLADI. BU SEYIRDE, KOMUTAN VE PILOT, NORMAL CALISMALARIN YANI SIRA, BIR COK SEYLE DE UGRASTILAR. BUNLAR UZAY TUTMASI, RADYO ARIZALARI, TIKANMIS TUVALET, LUMBUZLARDAKI KIRAGI, ARIZALI RADAR EKRANI VE UYKUSUZLUKTU. FAKAT HERSEYE KARSIN, COK BASARILI BIR SEYIRDI. ASTRONOTLAR, GEMININ SADECE BIR YUZUNU DAIMA GUNES'E CEVIREREK BIRKAC SAAT ISITTILAR, DOGAL OLARAK DIGER TARAF DA DONDU. BOYLECE GEMININ ISISAL OZELLIKLERI SAPTANMIS OLDU. MEKANIK KOLA YERLESTIRILEN BIR CIHAZLA, UZAY GEMISI CEVRESINDEKI PARCACIKLAR VE ELEKTRIK ALANLARI OLCULDU. MEKANIK KOLUN HAREKETINI SUREKLI DENETIM ALTINDA TUTMAK ICIN KOL UZERINE YERLESTIRILEN TELEVIZYON KAMERASI ARIZALANINCA, PERSONEL AYNI ISI YAPABILMEK ICIN BILDIGIMIZ AVCI DURBUNU KULLANMAK ZORUNDA KALDILAR. ILK UCUS GUNUNUN SONUNDA, YERYUZU'NDEN HAVALANIRKEN LUMBUZ KORUYUCUSUNU KIRAN BEYAZ MADDENIN, GEMININ BAS KISMINDAN KOPAN ISI KORUYUCU OLDUGUNU KESFETTILER. PERSONEL ILK GUN HICBIR SEY YIYEMEDI. AYRICA PILOT, AGIRLIKSIZ ORTAMA ALISAMADIGINDAN UYUYAMADI; DOLAYISIYLA DA IKINCI GUN COK YORGUN DUSMUSTU. BU DURUMU PILOT SU SOZLERLE DILE GETIRIYORDU: "KENDIMI, SANKI HER ON DAKIKADA BIR MARATON KOSUYORMUS GIBI HISSETTIM." BU SEYIRDE AYRICA ARI, PERVANE,

VE, SINEKLERDEN OLUSAN HAYVANLARIN, AGIRLIKSIZ ORTAMDA DAVRANISLARI INCELENDI. ARILAR UCMAKTAN YORULDUKLARINDA, AMACSIZ BIR SEKILDE OLDUKLARI YERE DONUYORLARDI. GEMI DUNYA'YA DONDUGUNDE TUM ARILAR OLMUSTU. PERVANELER CILGIN BIR SEKILDE KANAT CIRPTILAR; SINEKLER HEP YURUDULER. PILOT UCMAK ICIN CALISAN BIR SINEGI ASLA GORMEDIGINI SOYLUYORDU. INISIN YAPILACAGI EDWARDS HAVA KUVVETLERI USSU'NDEKI KURU GOL YATAGI MEVSIMIN DE ETKISIYLE INIS GUNU IYICE ISLANMISTI. BU NEDENLE, INIS ORAYA DEGIL DE, NEW MEXICO'DAKI LIMANA YAPILDI. FAKAT INISIN YAPILACAGI GUN KUVVETLI BIR FIRTINA PATLAMIS VE INISIN YAPILACAGI ALAN, SEYIRDEKI GEMIDEN DAHI RAHATCA GORULEBILINEN BEYAZ BIR TOZ BULUTU ALTINDA KALMISTI. BU NEDENLE UCUS BIR GUN GECIKTIRILDI. DORDUNCU UCUS, 27 HAZIRAN-4 TEMMUZ 1982 ARASI GERCEKLESTIRILDI. BU SEYIR DIGERLERINDEN IKI YONDEN FARKLIYDI. BIRINCISI, ASKERI AMACLI YUK TASIYORDU. HAVA KUVVETLERI YUKUN NE OLDUGUNU ACIKLAMADI. FAKAT BU GIZLI YUKUN, KIRMIZIOTESI ARAMA VE TARAMA YAPAN BIR ALET OLDUGU BILINIYORDU. IKINCI FARKLI YON, OGRENCILERIN HAZIRLADIGI 90 KG. AGIRLIGINDAKI DENEY PAKETININ TASINMASIYDI. BU SEYIRDE YAPILAN BIR BASKA DENEY DE BAZI BIYOLOJIK MATERYALIN BIRBIRLERINDEN AYRILMASIYDI. DENEYI YAPAN ALET, BU MATERYAL KARISIMI BIR ELEKTRIK ALANA KOYUYOR VE ONLARI DOGAL ELEKTRIK YUKLERINE GORE SECEBILIYORDU. DUNYA USTUNDE BU ISLEMI, YERCEKIMI ETKILEMEKTE ELEKTRIK YUKU, SICAKLIK VE CALKANTIYA NEDEN OLMAKTA, DOLAYISIYLA DA MATERYAL TEKRAR BIRBIRINE KARISMAKTADIR. UZAYDA BU MATERYALLERI BIRBIRINDEN AYIRMANIN, 800 KEZ DAHA ETKIN OLDUGU ORTAYA CIKARILDI. BU SON DENEME UCUSUYDU. BUNDAN SONRAKI UCUSLAR, NORMAL TICARI AMACLI OLACAKTI. DORDUNCU UCUSTA BASARIYA ULASAMAYAN EN ONEMLI NOKTA, KATI YAKITLI ROKETLERIN PARASUT MEKANIZMASININ ARIZALANMASI VE HER BIRI 7 MILYAR TL'NA MAL OLAN BU ROKETLERIN DENIZ DIBINI BOYLAMASIYDI. BESINCI UCUSUN PERSONEL SAYISI, ILK KEZ IKIDEN FAZLA OLUYORDU. UCUS KOMUTANI VE PILOTTAN BASKA, WILLIAM VE JOSEPH ADLI IKI ASTRONOT DA UCUS UZMANI OLARAK GEMIDE YER

ALDILAR. GEMININ ILK TICARI YUKU OLAN ILETISIM UYDULARI 11 KASIM 1982 GUNU BASLAYAN BU SEFERDE BASARIYLA YORUNGEYE OTURTULDU. EGER BU UYDULAR YERDEN YORUNGEYE YERLESTIRILSEYDI, UYDU SAHIPLERI DAHA FAZLA PARA ODEMEK ZORUNDA KALACAKLARDI. BU SEYIRDE PERSONELI UZAY TUTTU. BU YUZDEN UZAYDA YURUYUS IZLENCESI BIR GUN ERTELENDI. ERTESI GUN ISE HER BIRI YARIM MILYAR TL'NA MAL OLAN UZAY MELBUSATI ARIZALANDI. TUM UGRASLARA KARSIN ARIZALAR GIDERILEMEDIGI ICIN YURUYUSTEN VAZGECILDI. FAKAT BU COK ONEMLI BIR DENEYDI; CUNKI GELECEKTE UZAY LIMANI GIBI BUYUK YAPILAR INSA EDILIRKEN, BU TECHIZAT ILE ARAC DISI CALISMALAR YAPILACAK.

2. PROGRAMS

The two programs used to obtain the probabilities of the symbols in the magazine articles given above. A Fortran program creates a data set format which can be processed by a SAS program. The program which sets the logical record length of data file to 1, is given below.

```
//AKINSEL  JOB (0936,5555), 'AKINSEL',CLASS=A
//*MAIN ORG=NPGVM1.0936P
//  EXEC FORTVCG
//FORT.SYSIN DD*
C          THIS PROGRAM CONVERTS ONE LOGICAL RECORD OF
C          EIGHTY CHARACTERS TO EIGHTY
C          LOGICAL RECORDS OF ONE CHARACTER EACH.
C
C            UNIT 5: INPUT
C            UNIT 1: OUTPUT
C
      DIMENSION A(80)
      LINES =0
   10 CONTINUE
      READ(5.20,END=100) A
   20 FORMAT(80A1)
      LINES = LINES + 1
      DO 30 I=1.80
      WRITE(1,20) A(I)
   30 CONTINUE
      GO TO 10
  100 CONTINUE
      WRITE(6,110) LINES
  110 FORMAT(1X,'NUMBER OF LINES READ: ',17)
      STOP
      END
/*
//GO.FT01F001 DD UNIT=3350,VOL=SER=MVS004,
```

```
DISP=(NEW,KEEP),
//    DCB=(RECFM=FB,LRECL=,BLKSIZE=6000),
//    SPACE=(TRK,(1,1)),DSN=S0936.LETTER
//GO.SYSIN DD *
      Insert text here. (Also,remove this line).
/*
//
```

The second program is executed to find the probability of each symbol in the alphabet. This SAS program is given below.

```
//AKINSEL  JOB (0936,5555),'AKINSEL',CLASS=B
//*MAIN ORG=NPGVM1.0936P
//  EXEC SAS
//TEXT  DD UNIT=3350,VOL=SER=MVS004,DISP=SHR,
DSN=S0936.ALPHA1
//SYSIN DD *
OPTIONS LINESIZE = 80;
DATA TEXT;
    INFILE TEXT;
    INPUT  @1 LETTER $CHAR1. ;
    IF LETTER EQ ' ' THEN DELETE;
PROC FREQ DATA=TEXT;
    TABLES LETTER;
/*
//
```

APPENDIX B

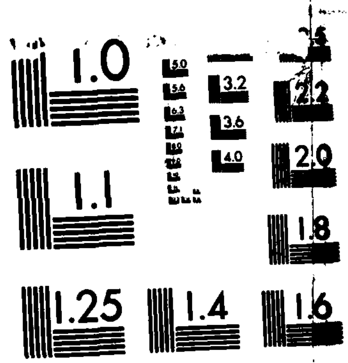
THE LISP PROGRAM OF CODING PROCESS

The Lisp program for finding the code words of the original Huffman and the modified Huffman codes is given below.

```
(defun huffman (P)
  (sortcar (assign (arrange (mapcar 'list P))) 'greaterp))

(defun arrange (Q)
  (cond ((null (cdr Q)) Q)
        (t (arrange (insert (list (add (caar Q) (caadr Q))
                                  (car Q) (cadr Q))
                            (cddr Q)) )) ))

(defun insert (x Q)
  (cond ((null Q) (cons x Q))
        ((lessp (plus (times (car x) K) epsilon) (caar Q))
        (putin N x Q))
        (t (cons (car Q) (insert x (cdr Q)) )) ))

(defun putin (n x L)
  (cond (( zerop n ) (cons x L))
        ((null L) (list x))
        (t (cons (car L) (putin (subl n) x (cdr L))))))

(defun assign (Q) (split nil (carQ)) )

(defun split (c L)
  (cond ((null (cdr L)) (list (list (car L) C)) )
        (t (append (split (cons 1 c) (cadı L))
                   (split (cons 0 c) (caddr L)) )) ))

(defun sortcode (L)
  (cond ((null L) nil)
   (t (inscode (caar L) (cadar L) (sortcode (cdr L)) )) ))
```

AD-A164 356 REDUCTION IN BANDWIDTH BY USING VARIABLE LENGTH CODES (U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA S AKINSEL DEC 85 2/2

UNCLASSIFIED F/G 9/4 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
(defun inscode (p c L)
  (cond ((null L) (list (list p c)) )
   ((greaterp (length c) (length (cadar L)))
   (cons (list p (cadar L)) (inscode (caar L) c (cdr L)) ))
  (t (cons (list p c) L)) ))

(defun totlength (L)
  (cond ((null L) 0)
        (t (add (times (caar L) (length (cadar L)) )
                (totlength (cdr L)) )) ))

(defun avglength (L)
  (quotient (times 1.0 (totlength L))
            (apply 'add (mapcar 'car L)) ))

(defun varlength (L)
  (quotient (times 1.0 (varlength2 L (avglength L)))
            (apply 'add (mapcar 'car L))))

(defun varlength2 (L mu)
  (cond ((null L) 0)
        (t (add (times (caar L)
              (expt (difference (length (cadar L)) mu) 2))
           (varlength2 (cdr L0 mu)))))

(defun Zipf (n)
  (cond ((zerop n) nil)
        (t (cons (quotient 1.0 n) (Zipf (- n 1)) )) ))

(defun tryN (n e k)
 (set 'N n)
 (set 'epsilon e)
 (set 'K k)
 (set code (sortcode (huffman Turkish)) )
 (print (list 'N '= n 'epsilon '= e 'K '= k))
 (pp code)
 (print (list 'mean '= (avglength code))) (terpr)
 (print (list 'variance '= (varlength code))) (terpr))
```

```
(set 'Turkish
'(0.0 0.00006 0.00006 0.00017 0.00028 0.00034
  0.00039 0.00045 0.00045 0.00056 0.00061 0.00067
  0.00067 0.00073 0.00073 0.00084 0.00084 0.00089
  0.00112 0.00134 0.00162 0.00196 0.00358 0.00581
  0.00687 0.00872 0.00989 0.01017 0.01224 0.01637
  0.01883 0.02185 0.02660 0.02682 0.02945 0.03213
  0.03509 0.03861 0.03984 0.05130 0.05163 0.06085
  0.06611 0.07952 0.09427 0.10528 0.13339))

(set 'N 0)
(set 'epsilon 0)
(set 'K 1)
```

APPENDIX C

THE FORTRAN PROGRAM TO FIND THE MAXIMUM BUFFER LENGTHS

```
$JOB
C
C     ***   VARIABLE DEFINITIONS   ***
C
C     LENGTH  = NUMBER OF BITS BELONGING TO EACH
C               CHARACTER AFTER CODING PROCESS
C     RATEI   = INPUT RATE (BITS PER UNIT TIME)
C     RATEO   = OUTPUT RATE (BITS PER UNIT TIME)
C     MAX     = MAXIMUM BUFFER LENGTH
C     BUF1(I) = BUFFER SIZE OF EACH CHARACTER
C     BUFFER  = TEMPORARY VARIABLE
C     BUF2    = REAL PART OF THE BUFFER
C     A(I)    = ARRAY IN WHICH THE CHARACTERS ARE LISTED
C     N(I)    = ARRAY IN WHICH THE NUMBER OF CHARACTERS
C               ARE LISTED
C
C     ***   VARIABLE DECLARATIONS   ***
C
      REAL BUFFER,LENGTH,RATEI,RATEO
      CHARACTER*1 A(     )
      INTEGER I,BUF1(     ),N(     ),MAX
C     (Insert the length of the messages inside
C      the parentheses given above.)
C
C     ***   BEGINNING OF THE PROGRAM   ***
C
      READ(5,100) A
      RATEI = (Insert the input rate.)
      PRINT,'INPUT RATE   IS  =   ',RATEI
      PRINT,' '
```

```
      RATEO = (Insert the output rate.)
      PRINT,'OUTPUT RATE  IS  =   ',RATEO
      PRINT,' '
      BUFFER   = 0.0
      DO 200 I = 1,(Insert the length of the message.)
         N(I)  = I
 200  CONTINUE
      WRITE(6,300)  'BUFFER SIZE'
C     (Insert the number of bits for each character
C      after coding process next to the variable
C      name 'LENGTH', given below.)
      DO 400 I = 1,(insert the length of the message)
         IF (A(I).EQ.'/')      THEN
            LENGTH =
         ELSE IF (A(I).EQ.'I') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'A') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'E') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'N') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'R') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'U') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'L') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'S') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'K') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'D') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'T') THEN
```

```
   LENGTH =
ELSE IF (A(I).EQ.'M') THEN
   LENGTH =
ELSE IF (A(I).EQ.'Y') THEN
   LENGTH =
ELSE IF (A(I).EQ.'O') THEN
   LENGTH =
ELSE IF (A(I).EQ.'G') THEN
   LENGTH =
ELSE IF (A(I).EQ.'B') THEN
   LENGTH =
ELSE IF (A(I).EQ.'C') THEN
   LENGTH =
ELSE IF (A(I).EQ.',') THEN
   LENGTH =
ELSE IF (A(I).EQ.'.') THEN
   LENGTH =
ELSE IF (A(I).EQ.'Z') THEN
   LENGTH =
ELSE IF (A(I).EQ.'V') THEN
   LENGTH =
ELSE IF (A(I).EQ.'P') THEN
   LENGTH =
ELSE IF (A(I).EQ.'H') THEN
   LENGTH =
ELSE IF (A(I).EQ.'F') THEN
   LENGTH =
ELSE IF (A(I).EQ.'O') THEN
   LENGTH =
ELSE IF (A(I).EQ.'''')THEN
   LENGTH =
ELSE IF (A(I).EQ.'1') THEN
   LENGTH =
ELSE IF (A(I).EQ.'"') THEN
   LENGTH =
```

```
ELSE IF (A(I).EQ.'2') THEN
   LENGTH =
ELSE IF (A(I).EQ.')') THEN
   LENGTH =
ELSE IF (A(I).EQ.'5') THEN
   LENGTH =
ELSE IF (A(I).EQ.'3') THEN
   LENGTH =
ELSE IF (A(I).EQ.'8') THEN
   LENGTH =
ELSE IF (A(I).EQ.'(') THEN
   LENGTH =
ELSE IF (A(I).EQ.'4') THEN
   LENGTH =
ELSE IF (A(I).EQ.';') THEN
   LENGTH =
ELSE IF (A(I).EQ.'9') THEN
   LENGTH =
ELSE IF (A(I).EQ.'J') THEN
   LENGTH =
ELSE IF (A(I).EQ.'6') THEN
   LENGTH =
ELSE IF (A(I).EQ.'W') THEN
   LENGTH =
ELSE IF (A(I).EQ.':') THEN
   LENGTH =
ELSE IF (A(I).EQ.'7') THEN
   LENGTH =
ELSE IF (A(I).EQ.'-') THEN
   LENGTH =
ELSE IF (A(I).EQ.'?') THEN
   LENGTH =
ELSE IF (A(I).EQ.'X') THEN
   LENGTH =
ELSE IF (A(I).EQ.'Q') THEN
```

```
               LENGTH =
            END IF
            BUFFER = BUFFER + LENGTH
            BUFFER = BUFFER - RATEO
            IF (BUFFER.LE.0.0) THEN
               BUFFER  = 0.0
               BUF1(I) = BUFFER
            ELSE
               BUF2 = BUFFER - AINT(BUFFER)
               IF (BUF2.GT.0.0) THEN
                  BUF1(I) = INT(BUFFER)
                  BUF1(I) = BUF1(I) + 1
               ELSE
                  BUF1(I) = INT(BUFFER)
               END IF
            END IF
 400     CONTINUE
         MAX = 0
         DO 500 I = 1,(Insert the length of the message.)
            IF (MAX.LT.BUF1(I)) THEN
               MAX = BUF1(I)
            END IF
 500     CONTINUE
         DO 600 I = 1,(Insert the length of the message.)
            WRITE(6,700) N(I),BUF1(I)
 600     CONTINUE
 100     FORMAT(73 A1)
 700     FORMAT (I6,9X,I5)
 300     FORMAT (9X,A12)
         STOP
         END
$ENTRY
C        (Insert the message itself below. Each line
C         should consist of 73 characters.)
```

APPENDIX D

FORTRAN PROGRAM TO FIND THE NUMBER OF OVERFLOWS

```
$JOB
C
C      ***   VARIABLE DEFINITIONS   ***
C
C      LENGTH = NUMBER OF BITS BELONGING TO EACH
C               CHARACTER AFTER ENCODING PROCESS
C      RATEI  = INPUT RATE (BITS PER UNIT TIME)
C      RATEO  = OUTPUT RATE (BITS PER UNIT TIME)
C      P      = NUMBER OF LOST CHARACTERS
C      R      = NUMBER OF TRANSMITTED CHARACTERS
C      DIFFER = DIFFERENCE BETWEEN LENGTH OF THE CHARACTER
C               AND OUTPUT RATE
C      BUFFER = PROVIDED BUFFER SIZE
C      A(I)   = ARRAY IN WHICH THE CHARACTERS ARE LISTED
C      DIFF1  = INTEGER PART OF DIFFER
C      DIFF2  = REAL PART OF DIFFER
C      FLOW   = DIFFERENCE BETWEEN BUFFER AND DIFF1
C
C      ***   VARIABLE DECLARATIONS   ***
C
       REAL LENGTH,RATEI,RATEO,DIFFER,DIFF2
       CHARACTER*1 A(      )
C      (Insert the length of the message inside the
C      parenthesis given above.)
       INTEGER I,K,P,BUFFER,R,DIFF1,FLOW,T
C
C      ***   BEGINNING OF THE PROGRAM   ***
C
       READ(5,100) A
       DO 200 T  =(    ,   )
```

```
C     (Insert the smallest and the largest provided
C      buffer size in the parenthesis above.)
        BUFFER = T
        P = 0
        R = 0
        DIFFER = 0.0
       RATEI = (Insert the input rate.)
       PRINT,'INPUT RATE  IS  =   ',RATEI
       PRINT,' '
       RATEO = (Insert the output rate.)
       PRINT,'OUTPUT RATE IS  =   ',RATEO
       PRINT,' '
       PRINT,'PROVIDED BUFFER  IS  =   ',BUFFER
       PRINT,' '
C     (Insert the number of bits for each character
C      after coding process next to the variable
C      name 'LENGTH', given below.)
       DO 300 I= 1,(Insert the length of the message.)
          IF (A(I).EQ.'/')      THEN
             LENGTH =
          ELSE IF (A(I).EQ.'I') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'A') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'E') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'N') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'R') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'U') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'L') THEN
             LENGTH =
          ELSE IF (A(I).EQ.'S') THEN
```

```
   LENGTH =
ELSE IF (A(I).EQ.'K') THEN
   LENGTH =
ELSE IF (A(I).EQ.'D') THEN
   LENGTH =
ELSE IF (A(I).EQ.'T') THEN
   LENGTH =
ELSE IF (A(I).EQ.'M') THEN
   LENGTH =
ELSE IF (A(I).EQ.'Y') THEN
   LENGTH =
ELSE IF (A(I).EQ.'O') THEN
   LENGTH =
ELSE IF (A(I).EQ.'G') THEN
   LENGTH =
ELSE IF (A(I).EQ.'B') THEN
   LENGTH =
ELSE IF (A(I).EQ.'C') THEN
   LENGTH =
ELSE IF (A(I).EQ.',') THEN
   LENGTH =
ELSE IF (A(I).EQ.'.') THEN
   LENGTH =
ELSE IF (A(I).EQ.'Z') THEN
   LENGTH =
ELSE IF (A(I).EQ.'V') THEN
   LENGTH =
ELSE IF (A(I).EQ.'P') THEN
   LENGTH =
ELSE IF (A(I).EQ.'H') THEN
   LENGTH =
ELSE IF (A(I).EQ.'F') THEN
   LENGTH =
ELSE IF (A(I).EQ.'O') THEN
   LENGTH =
```

```
ELSE IF (A(I).EQ.'''') THEN
   LENGTH =
ELSE IF (A(I).EQ.'1') THEN
   LENGTH =
ELSE IF (A(I).EQ.'"') THEN
   LENGTH =
ELSE IF (A(I).EQ.'2') THEN
   LENGTH =
ELSE IF (A(I).EQ.')') THEN
   LENGTH =
ELSE IF (A(I).EQ.'5') THEN
   LENGTH =
ELSE IF (A(I).EQ.'3') THEN
   LENGTH =
ELSE IF (A(I).EQ.'8') THEN
   LENGTH =
ELSE IF (A(I).EQ.'(') THEN
   LENGTH =
ELSE IF (A(I).EQ.'4') THEN
   LENGTH =
ELSE IF (A(I).EQ.';') THEN
   LENGTH =
ELSE IF (A(I).EQ.'9') THEN
   LENGTH =
ELSE IF (A(I).EQ.'J') THEN
   LENGTH =
ELSE IF (A(I).EQ.'6') THEN
   LENGTH =
ELSE IF (A(I).EQ.'W') THEN
   LENGTH =
ELSE IF (A(I).EQ.':') THEN
   LENGTH =
ELSE IF (A(I).EQ.'7') THEN
   LENGTH =
ELSE IF (A(I).EQ.'-') THEN
```

```
            LENGTH =
         ELSE IF (A(I).EQ.'?') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'X') THEN
            LENGTH =
         ELSE IF (A(I).EQ.'Q') THEN
            LENGTH =
         END IF
         LENGTH = LENGTH + DIFFER
         DIFFER = LENGTH - RATEO
         IF (DIFFER.LE.0.0) THEN
            R = R + 1
            DIFFER = 0.0
         ELSE
            DIFF2 = DIFFER - AINT(DIFFER)
            IF (DIFF2.GT.0.0) THEN
               DIFF1 = INT(DIFFER)
               DIFF1 = DIFF1 + 1
            ELSE
               DIFF1 = INT(DIFFER)
            END IF
            FLOW  = BUFFER - DIFF1
            IF (FLOW.LT.0)THEN
               P = P + 1
            ELSE
               R = R + 1
            END IF
         END IF
300   CONTINUE
      PRINT,'LOST CHARACTERS        ARE  = ',P
      PRINT,'TRANSMITTED CHARACTERS ARE  = ',R
      PRINT,' '
      PRINT,' '
      PRINT,' '
200   CONTINUE
```

```
 100  FORMAT(73 A1)
      STOP
      END
$ENTRY
C      (Insert the message itself below. Each line
C       should consist of 73 characters.)
```

LIST OF REFERENCES

1. Hamming, R.W., Coding and Information Theory, Prentice-Hall, Inc., 1980.

2. Huffman, D., "A Method for the Construction of Minimum Redundancy Codes", Proceedings of the Institude of Radio Engineers, Vol. 40, pp. 1098-1101, September 1952.

3. Kilic, Suha , Modification of Huffman Coding, Master's Thesis, Naval Postgraduate School, Monterey, CA., March 1985.

4. Stegers, Wolfgang, ceyiren Hataysal H. "Modern Gemilerin Garip Bicimleri", Bilim ve Teknik, Cilt 16, Sayi 191, Ekim 1983.

5. Dr. Derman I. Ethem, "Uzay Mekigi'nin Oykusu", Bilim ve Teknik, Cilt 17, Sayi 194, Ocak 1984.

6. SAS Institute Inc. SAS User's Guide: Basics 1982 Edition, Cary NC: SAS Institute Inc., 1982.

7. Foderaro, John K., The Franz Lisp Manual, University of California, 1980.

8. Winston, Patrick Henry, Horn Berthold Klaus Paul, LISP, 1984.

9. Kleinrock, Leonard, Communication Nets, Dover Publications, Inc., 1972.

10. Kleinrock, Leonard, Queueing Systems, Volume II, Computer Applications, John Wiley & Sons, Inc., 1976.

BIBLIOGRAPHY

Allen, O. Arnold, Probability, Statistics, and Queueing Theory, with Computer Science Applications, Academic Press, 1978.

Riordan, John, Stochastic Service Systems, John Wiley & Sons, Inc., 1962.

Trivedi, S. Kishor, Probability & Statistics with Reliability Queueing and Computer Science Applications, Prentice-Hall, Inc., 1982.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Department Chairman, Code 62Rr Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
4.	Department Chairman, Code 52MI Department of Computer Science Naval Postgraduate School Monterey, California 93943	1
5.	Prof. Hamming R.W., Code 52Hg Department of Computer Science Naval Postgraduate School Monterey, California 93943	3
6.	Prof. Fredricksen H., Code 53Fs Department of Mathematics Naval Postgraduate School Monterey, California 93943	1
7.	Prof. Jin-Fu Chang Department of Electrical Engineering National Taiwan University Taipei-Taiwan 107 REPUBLIC OF CHINA	1
8.	Serdar Akinsel Ethemefendi Cad. Abdulhalikrenda Sok. Meral Apt. No:12 D:16 Erenkoy, Istanbul TURKEY	4
9.	Deniz Kuvvetleri Komutanligi Personel Daire Baskanligi Bakanliklar, Ankara TURKEY	5
10.	Deniz Harb Okulu Komutanligi Fen Bilimleri Bolum Baskanligi Tuzla, Istanbul TURKEY	1
11.	Deniz Harb Okulu Komutanligi Kutuphanesi Tuzla, Istanbul TURKEY	1
12.	Kara Harb Okulu Komutanligi Kutuphanesi Bakanliklar, Ankara TURKEY	1
13.	Hava Harb Okulu Komutanligi Kutuphanesi Yesilyurt, Istanbul TURKEY	1

14. Istanbul Teknik Universitesi
Elektrik Fakultesi Kutuphanesi
Istanbul, TURKEY — 1

15. Bogazici Universitesi
Elektrik Fakultesi Kutuphanesi
Istanbul, TURKEY — 1

16. Orta Dogu Teknik Universitesi
Elektrik Fakultesi Kutuphanesi
Ankara, TURKEY — 1

17. Abraham Hazbun
244 Van Buren
Monterey, California 93940 — 1

18. Zafer Betoner
Ahmet Mithat Efendi Cad. No:4/1
Fenerbahce, Istanbul TURKEY — 1

END

FILMED

3-86

DTIC